RSAC 2025 Conference

Many Voices.
One Community.

SESSION ID: TPV-M01

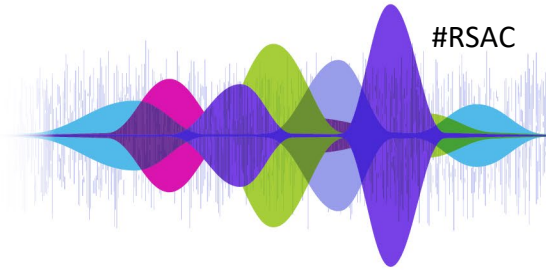
Cracks in the Fortress: How Major Companies Leak Their Secrets

Yakir Kadkoda

Director of Security Research
Aqua Security
linkedin.com/in/yakirkad/

Assaf Morag

Director of Threat Intelligence
Aqua Security
linkedin.com/in/assafmo/



Disclaimer

Presentations are intended for educational purposes only and do not replace independent professional judgment. Statements of fact and opinions expressed are those of the presenters individually and, unless expressly stated to the contrary, are not the opinion or position of RSA Conference LLC or any other co-sponsors. RSA Conference LLC does not endorse or approve, and assumes no responsibility for, the content, accuracy or completeness of the information presented.

Attendees should note that sessions may be audio- or video-recorded and may be published in various media, including print, audio and video formats without further notice. The presentation template and any media capture are subject to copyright protection.

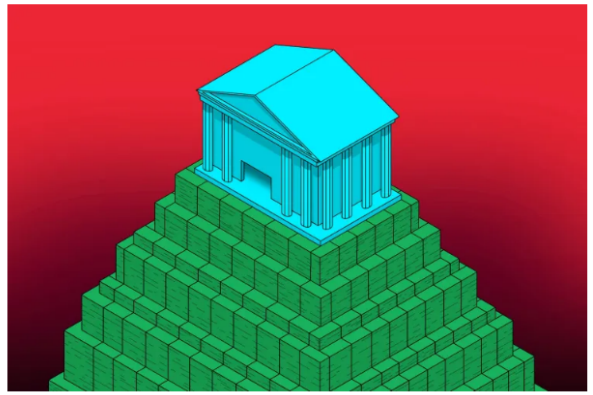
© 2025 RSA Conference LLC or its affiliates. The RSAC and RSAC CONFERENCE logos and other trademarks are proprietary. All rights reserved.

COMPLIANCE FRAMEWORKS
ADVANCED THREAT DETECTION
ZERO TRUST ARCHITECTURE
PROACTIVE RED TEAMING
AI-DRIVEN SOC
HARDCODED SECRET
IN A PUBLIC REPO
imgflip.com

Our Research Mindset

NEWS

The US Treasury Department was hacked



/ The Treasury Department said a China-based threat actor gained access to several employee workstations and unclassified documents.

by Emma Roth
Dec 31, 2024, 12:25 AM GMT+2

25 Comments (25 New)

Dropbox Breach Exposes Customer Credentials, Authentication Data

Threat actor dropped in to Dropbox Sign production environment and accessed emails, passwords, with APIs, OAuth, and MFA info.

Elizabeth Montalbano, Contributing Writer
May 2, 2024

3 Min Read

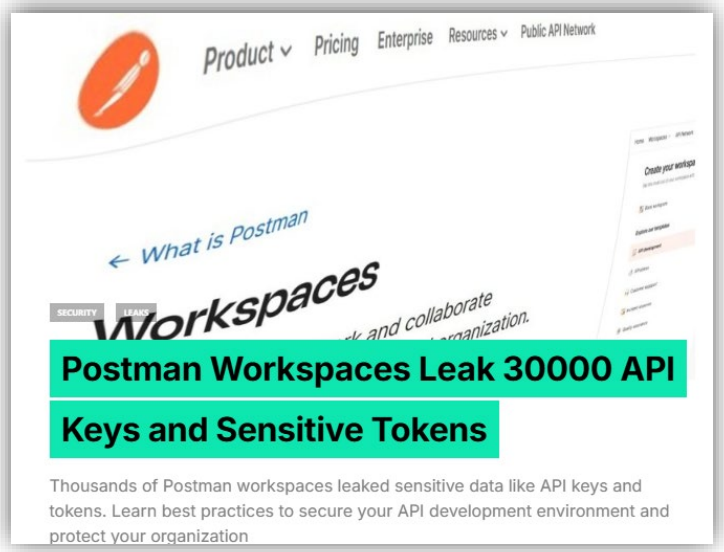
Editor's Cho

DATA BREACHES

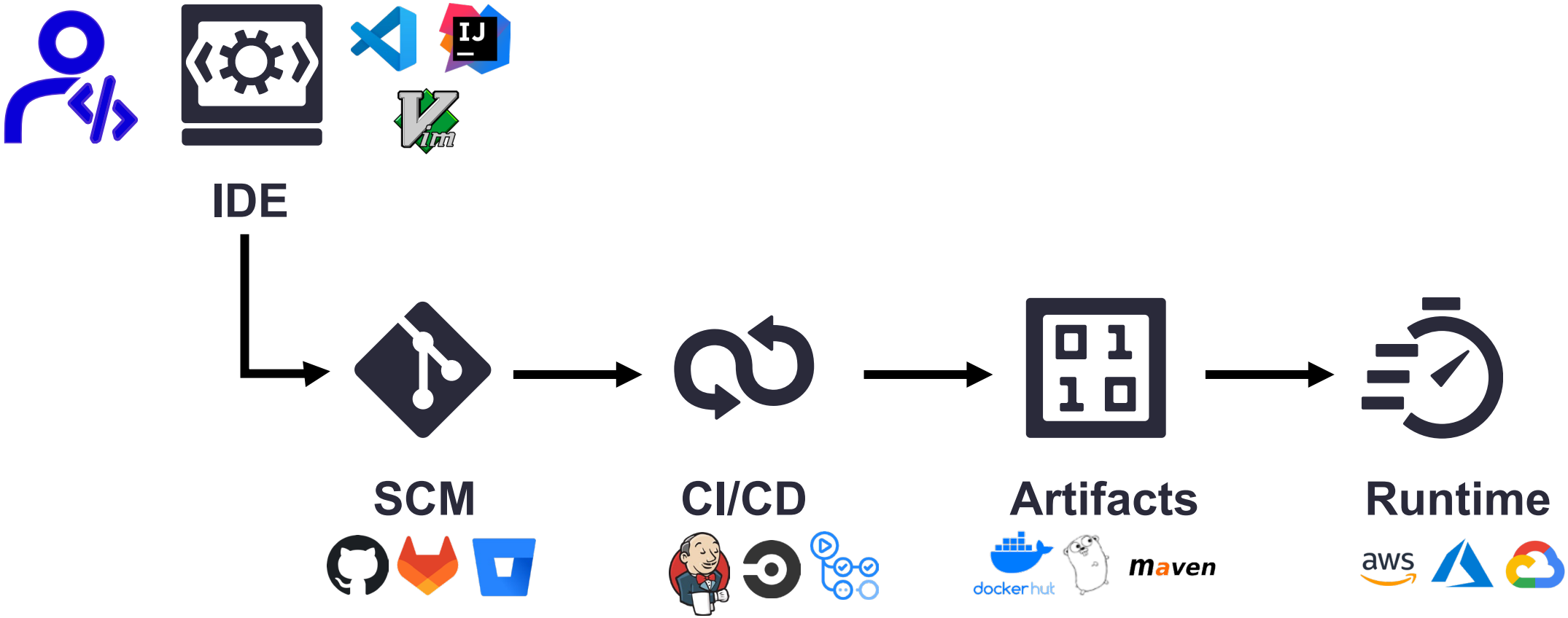
New York Times Responds to Source Code Leak

The New York Times has issued a statement after someone leaked source code allegedly belonging to the news giant.

By Eduard Kovacs
June 10, 2024


Product
Pricing
Enterprise
Resources
Public API Network
What is Postman
Workspaces
Postman Workspaces Leak 30000 API Keys and Sensitive Tokens
Thousands of Postman workspaces leaked sensitive data like API keys and tokens. Learn best practices to secure your API development environment and protect your organization

The Development Flow



The Development Flow

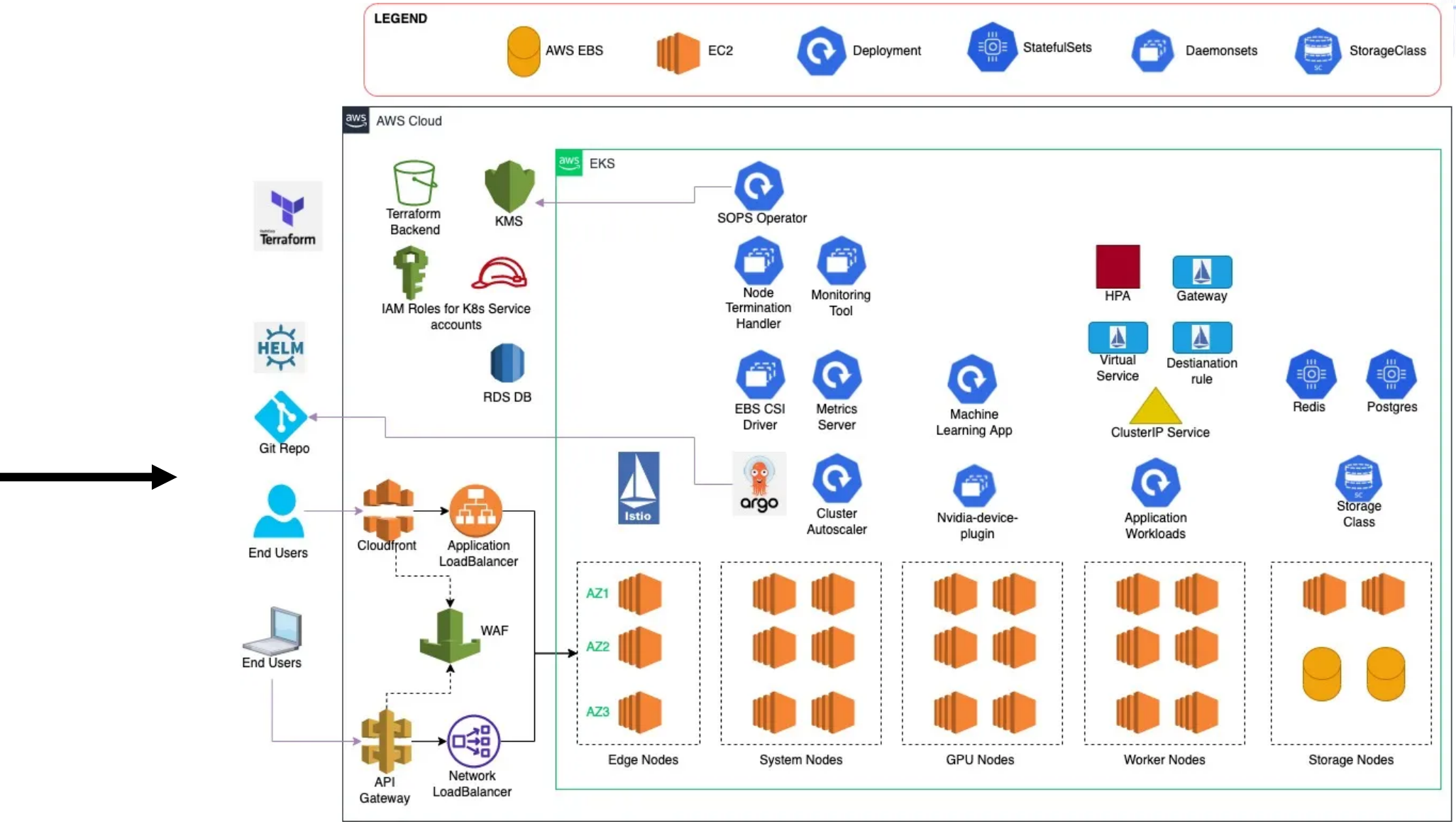
IDE

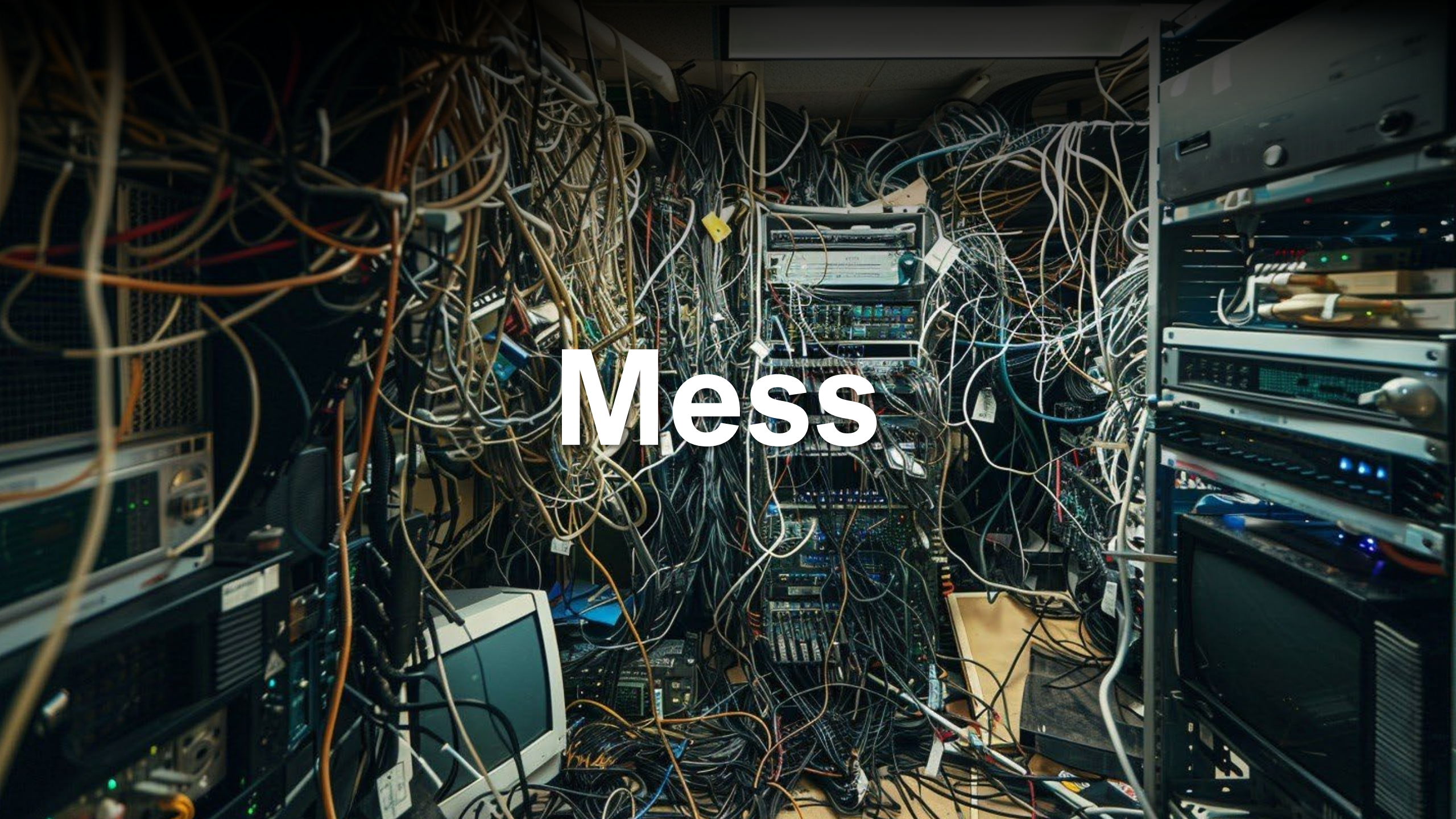
SCM

CI/CD

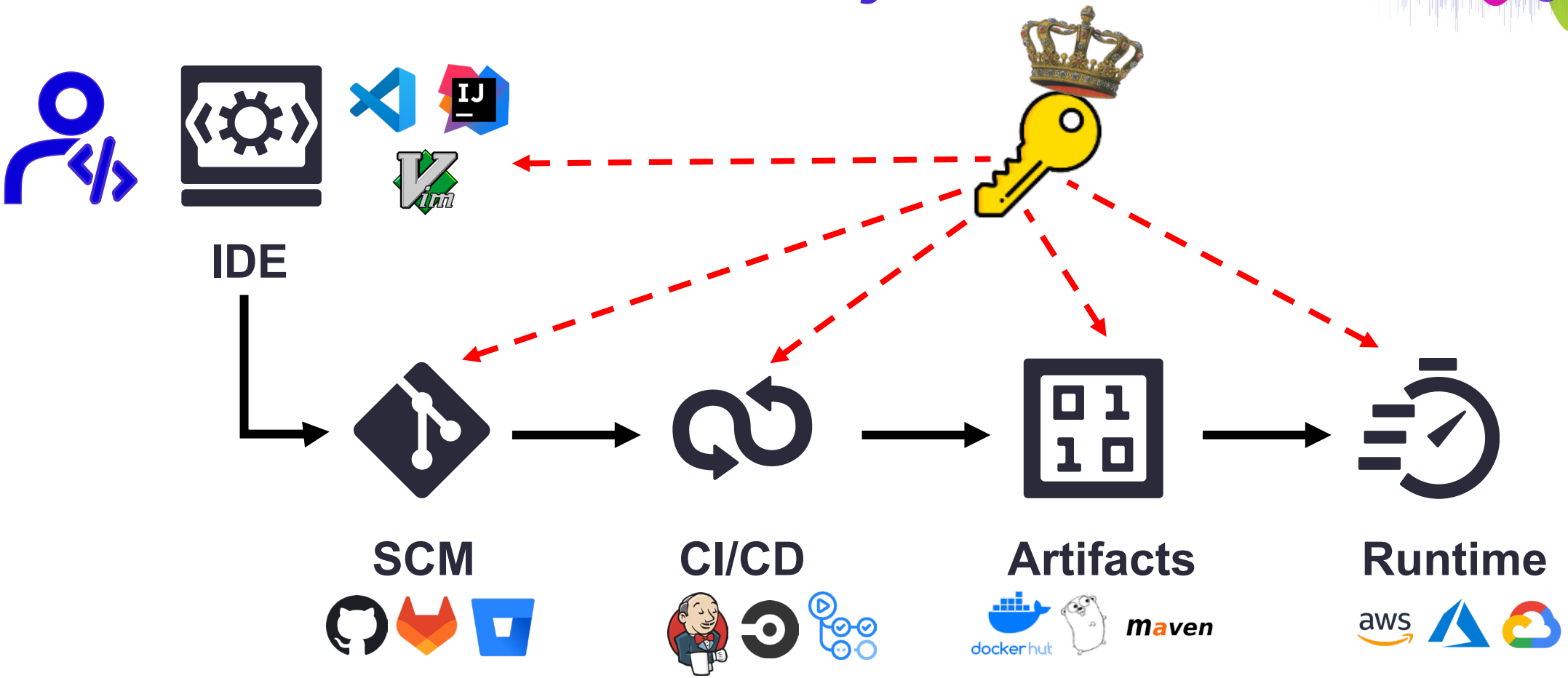
Artifacts

Runtime


LEGEND
AWS EBS
EC2
Deployment
StatefulSets
Daemonsets
StorageClass
AWS Cloud
EKS
Terraform
Terraform Backend
KMS
SOPS Operator
IAM Roles for K8s Service accounts
Node Termination Handler
Monitoring Tool
HPA
Gateway
HELM
RDS DB
Virtual Service
Destianation rule
EBS CSI Driver
Metrics Server
Machine Learning App
Redis
Postgres
ClusterIP Service
Git Repo
Istio
argo
Cluster Autoscaler
Nvidia-device-plugin
Application Workloads
Storage Class
End Users
Cloudfront
Application LoadBalancer
WAF
End Users
AZ1
AZ2
AZ3
API Gateway
Network LoadBalancer
Edge Nodes
System Nodes
GPU Nodes
Worker Nodes
Storage Nodes


Mess

Secrets Can Leak from Any Phase



Key Topics for Discussion

Kubernetes Secrets
Secrets in Registries
Phantom Secrets
Shadow-IT Secrets
Secrets in Applications

RSAC™ 2025 Conference

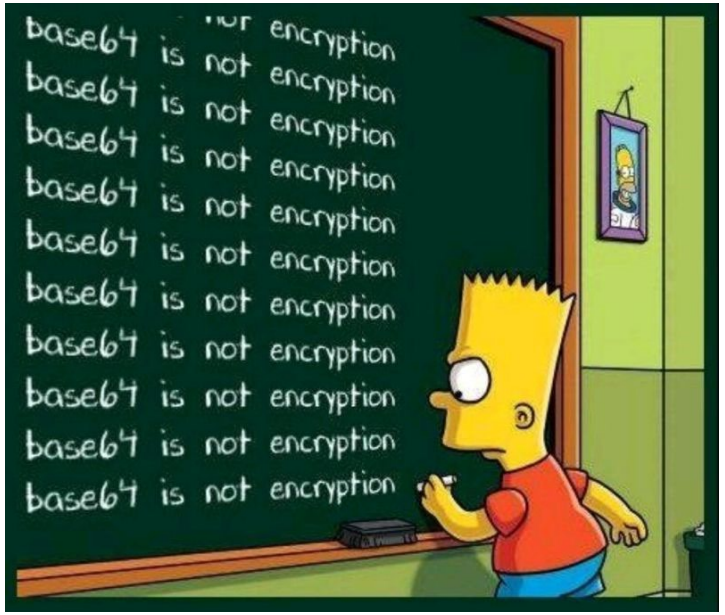
Kubernetes Secrets

Many Voices.
One Community.

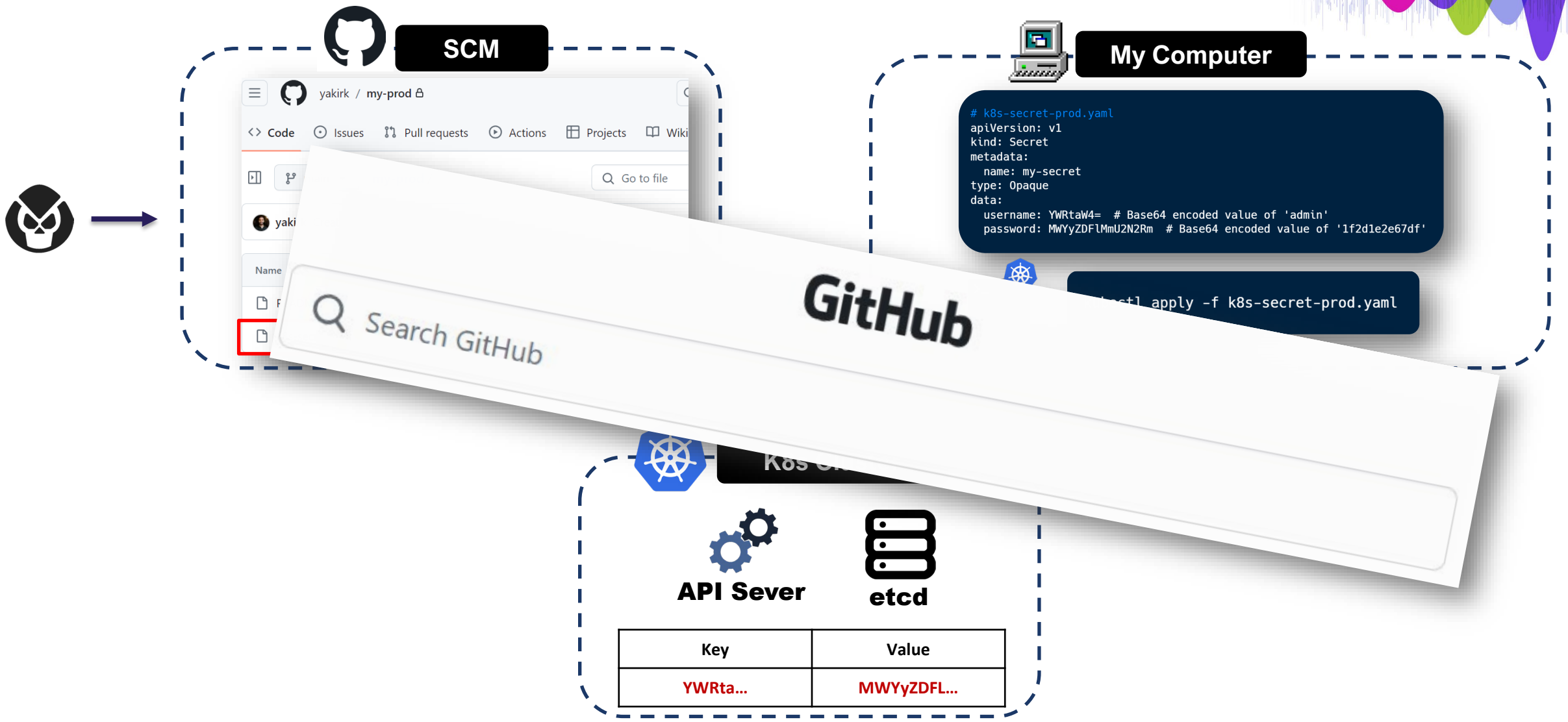
What Are K8s Secrets?

```yaml
# k8s-secret-prod.yaml
apiVersion: v1
kind: Secret
metadata:
  name: my-secret
type: Opaque
data:
  username: YWRtaW4=  # Base64 encoded value of 'admin'
  password: MWYyZDFlMmU2N2Rm  # Base64 encoded value of '1f2d1e2e67df'
```

Built-in Type	Usage
Opaque	arbitrary user-defined data
kubernetes.io/service-account-token	ServiceAccount token
kubernetes.io/dockercfg	serialized ~/.dockercfg file
kubernetes.io/dockerconfigjson	serialized ~/.docker/config.json file
kubernetes.io/basic-auth	credentials for basic authentication
kubernetes.io/ssh-auth	credentials for SSH authentication
kubernetes.io/tls	data for a TLS client or server
bootstrap.kubernetes.io/token	bootstrap token data



What Are K8s Secrets?



The Findings

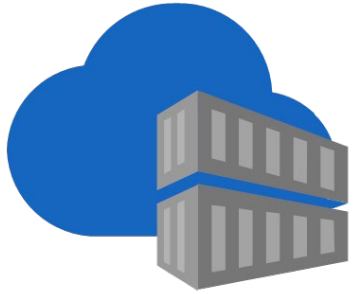
kubernetes.io/dockercfg	serialized ~/.dockercfg file
kubernetes.io/dockerconfigjson	serialized ~/.docker/config.json file

```yaml
# An example of exposed encoded secret in secret.yaml
apiVersion: v1
kind: Secret
type: kubernetes.io/dockerconfigjson
metadata:
  name: image-pull-push-secret-src
  namespace: imagepullpushsecret-patcher
data:
  .dockerconfigjson:
eyJhdXRocyI6eyJteXJlZ2lzdHJ5LmNvbTo1MDAwIjp7IlVzZXJuYW1lIjoidGVzdC
IsIlBhc3N3b3JkIjoiMTIzNDU2IiwiRW1haWwiOiJteUBlbWFpbC5jb20ifX19=
```

```json
# An example of exposed decoded secret
{
    "auths":
    {
        "myregistry.com:5000":
        {
            "Username": "test",
            "Password": "123456",
            "Email": "my@email.com"
        }
    }
}
```

/(dockerconfigjson|dockercfg):\s*\|*\s*(ey|ew)+/ path:*.yml

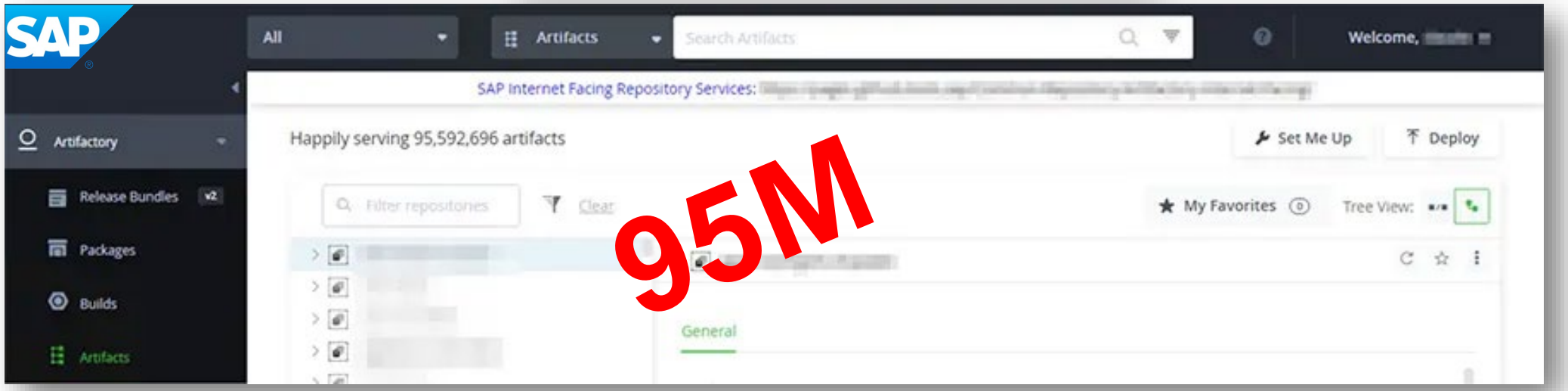
Harvesting “dockercfg”



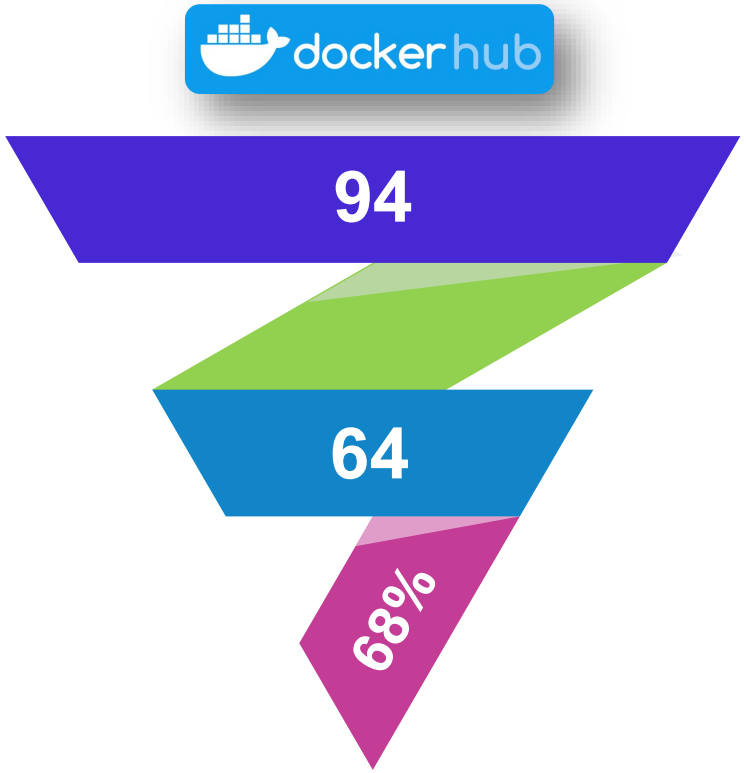
46.3%

Use Case 1: Artifacts Repository

```
apiVersion: v1
kind: Secret
metadata:
  name: docker-registry
data:
  .dockerconfigjson: ewogICJh*******
```



Use Case 2: Docker Hub Accounts



#	Total Pull Count	Total Container Images
1	21,232,822	238
2	13,037,290	100
3	3,438,112	379
4	2,444,652	16
5	2,400,573	45

Mitigations for K8s Secrets

Most secret scanning tools do not alert or check for encoded secrets

Scan your codebase for Kubernetes Secrets

Ensure you encrypt data uploaded to SCM with tools like **Sealed Secrets** or **Mozilla SOPS**

docker-config-secret.yaml

```
rules:
  - id: rule_dockerconfig
    category: general
    title: Dockerconfig secret exposed
    severity: HIGH
    regex: (?i)(\.(dockerconfigjson|dockercfg):\s*\|*\s*(ey|ew)+)
```

```
apiVersion: bitnami.com/v1alpha1
kind: SealedSecret
metadata:
  name: mysecret
  namespace: mynamespace
spec:
  encryptedData:
    foo: AgBy3i4OJSWK+PiTySYZZA9rO43cGDEq.....
```

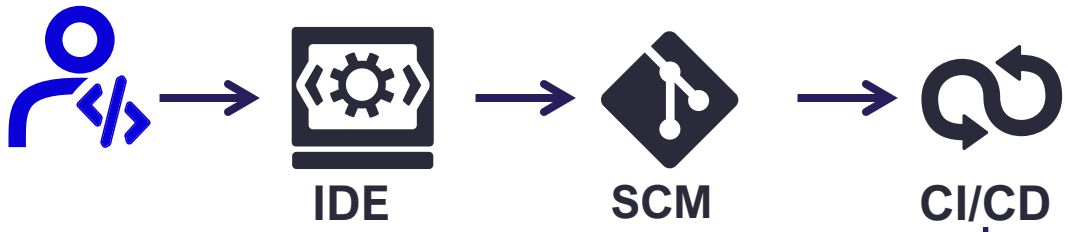
https://github.com/bitnami-labs/sealed-secrets
https://github.com/getsops/sops

RSAC™ 2025 Conference

Secrets in Registries

Many Voices.
One Community.

What Are Registries?



Upload

GCP GCR

JFrog Artifactory

Azure ACR

nexus repository

AWS ECR

Consumers

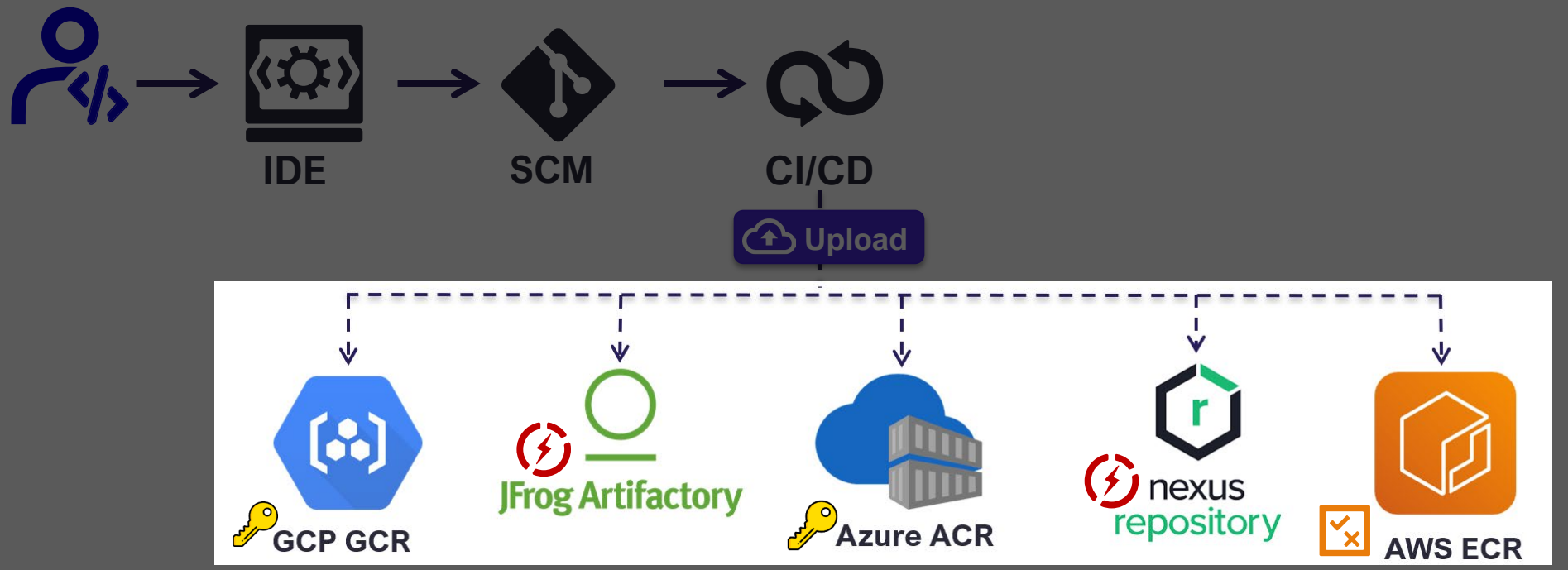
Internal User

External User

Runtime

Download

What Are Registries?



Consumers

Internal User

External User

Runtime

Download

Use Cases

// 5000 / TCP

Docker Registry HTTP API

```
HTTP/1.1 200 OK
Cache-Control: no-cache
Date: Sun, 04 Feb 2024 13:11:41 GMT
Content-Length: 0


Docker Registry HTTP API:
  Repositories:
    api-gateway
    ceylon-bank-api
    ceylon-bank-web
    insightdb-eureka-server
    oss-application-token-service
    process-flow
```

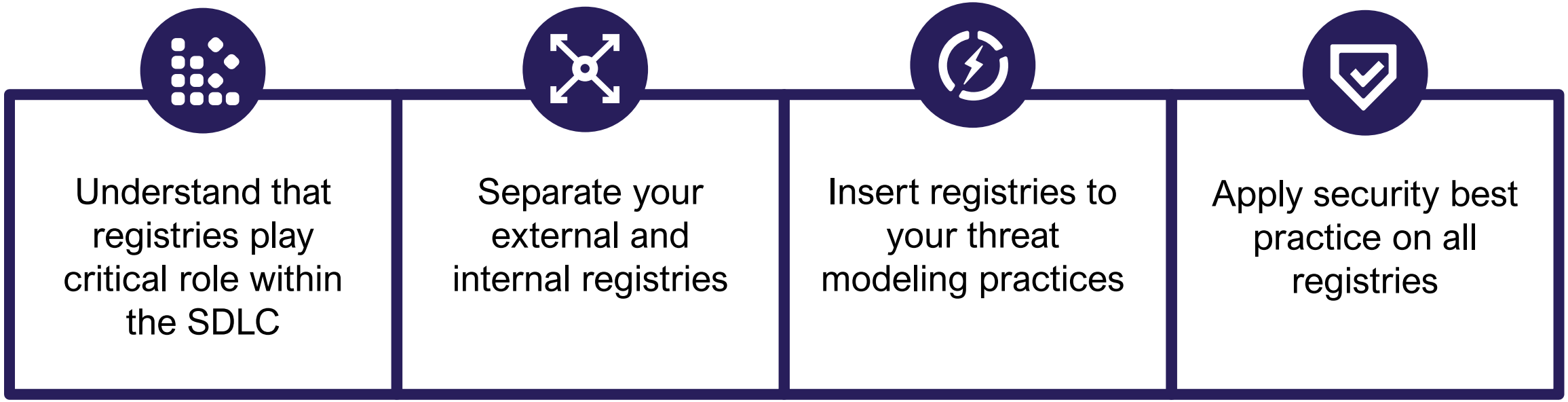
Builds › common-logic › 1669025386246

Build Name	Agent	Build Agent	Started	Duration	Principal	Artifac
common-logic	Gradle/6.1.1	Gradle/6.1.1	21-11-22 12:09:46 +...	1.9 seconds	distiller	admin

buildInfo.env.APP_NAME_1933	Gradle
buildInfo.env.artifactory_password	[redacted]
buildInfo.env.artifactory_user	admin
buildInfo.env.AWS_ACCESS_KEY_ID	[redacted]
buildInfo.env.AWS_SECRET_ACCESS_KEY	[redacted]
buildInfo.env.BASH_ENV	/tmp/.bash_env-637b4c7d28af1655ba05cff8-0-build

38.7%

Mitigations for Secrets in Registries

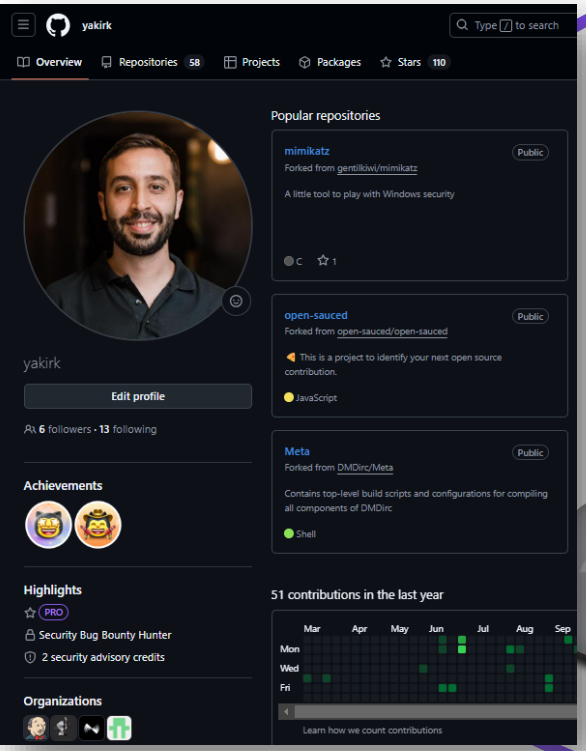

Understand that registries play critical role within the SDLC
Separate your external and internal registries
Insert registries to your threat modeling practices
Apply security best practice on all registries

RSAC™ 2025 Conference

Shadow-IT Secrets

Many Voices.
One Community.

What is Shadow-IT?



GitLab

slack

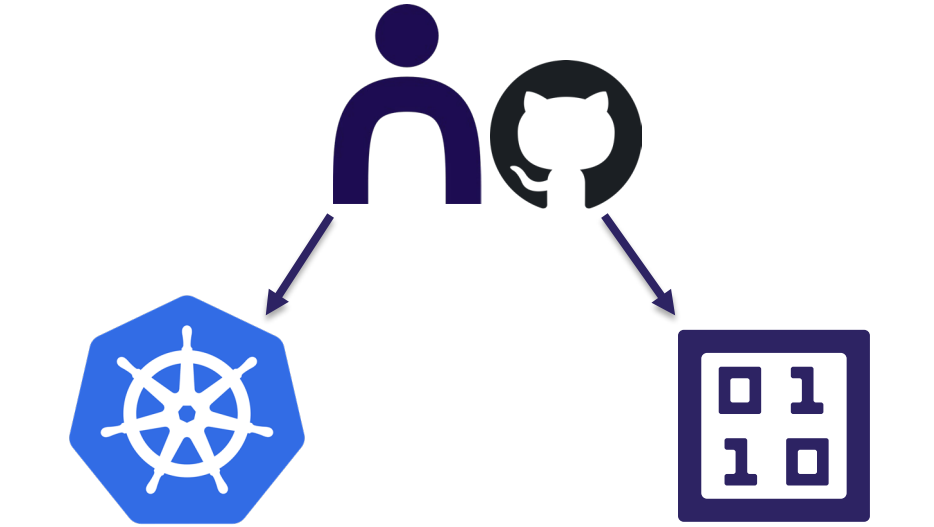
zoom

Approved IT

Office 365

Shadow-IT

Statistics



Kubernetes Secrets

66%

Secrets in Registries

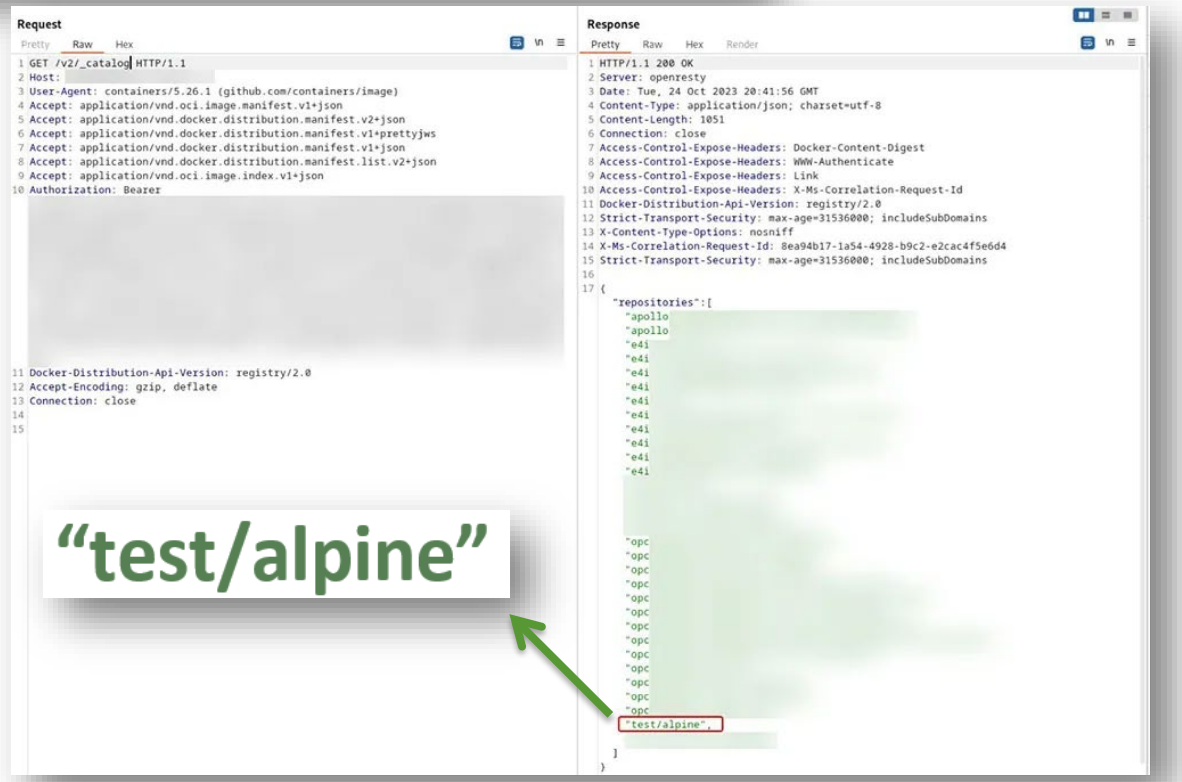
83%

Use Case : Azure Internal Projects

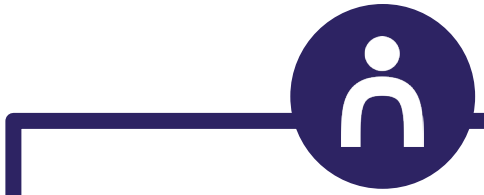
```
└─# docker login                    ..azurecr.io -u                  -p "K9f                    cDD"
WARNING! Using --password via the CLI is insecure. Use --password-stdin.
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

Internal ACR that include images of projects like **Azure IoT Edge**, **Akiri**, etc.



Mitigations for Shadow-IT



Encourage employees to **scan their personal accounts** for secrets

Scan GitHub/Internet for unique keys and exposed URLs like ‘myorg.com/artifacts'

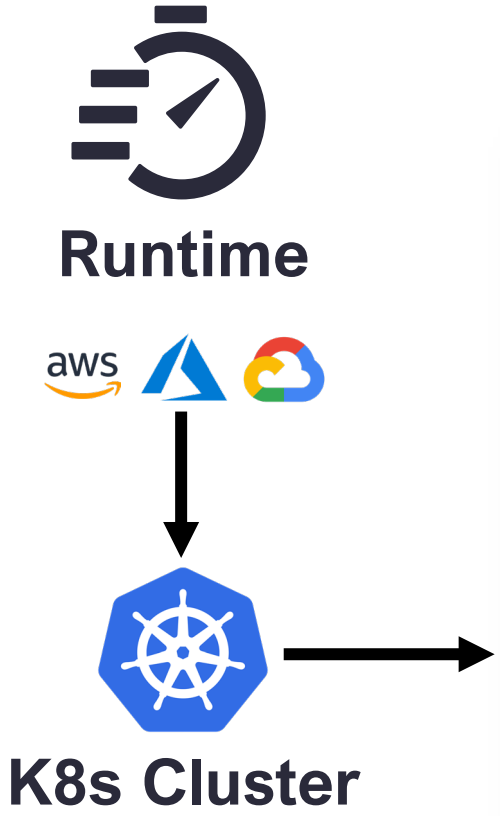
Limit keys to only necessary privileges and scope

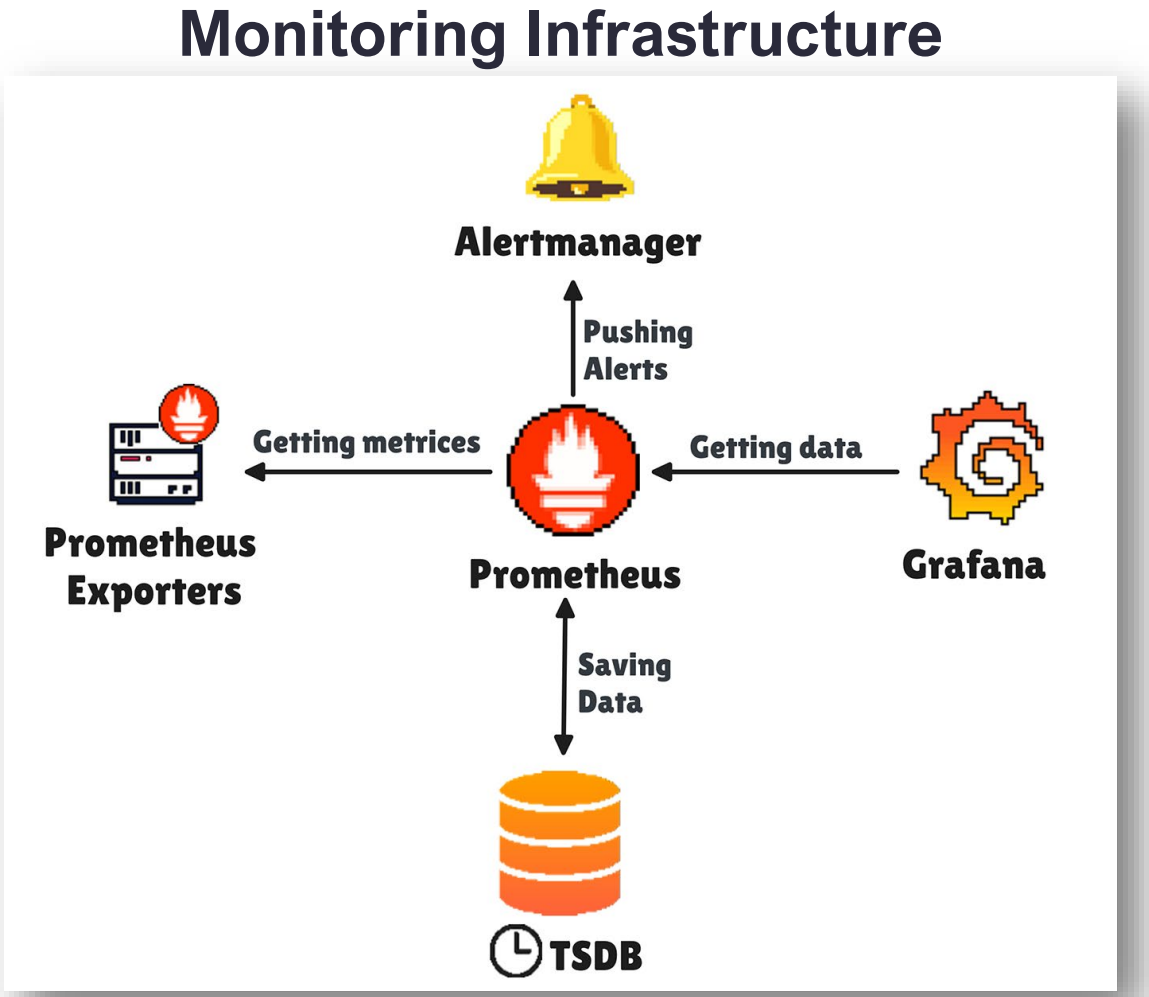
RSAC™ 2025 Conference

Secrets in Applications

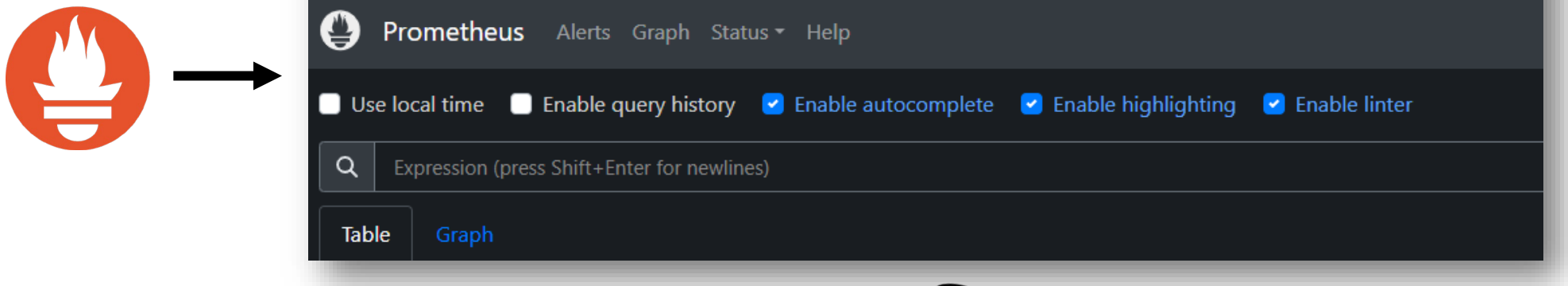
Many Voices.
One Community.

Runtime Applications



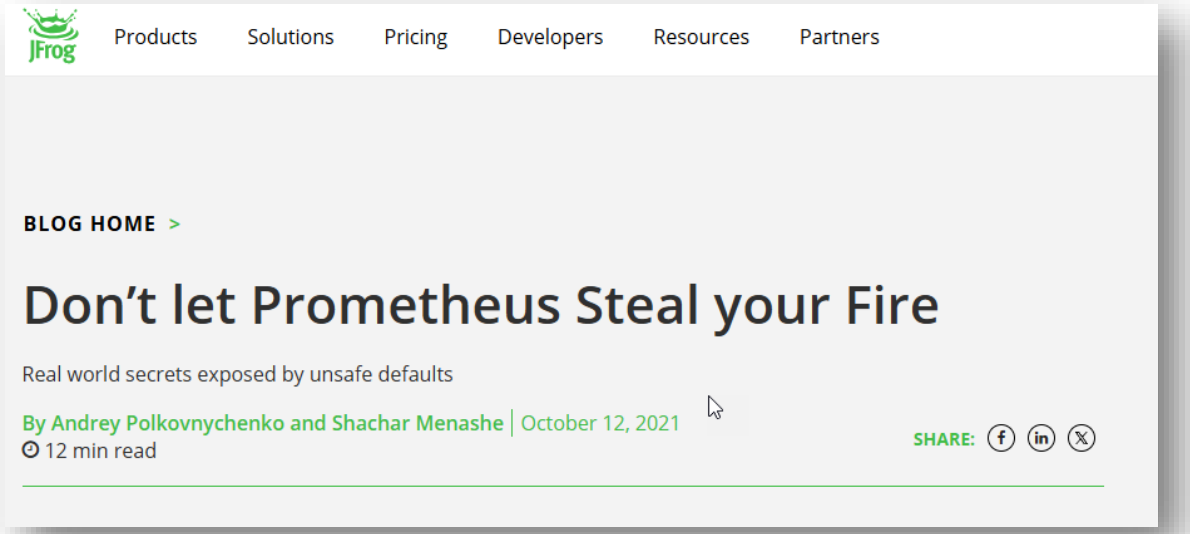


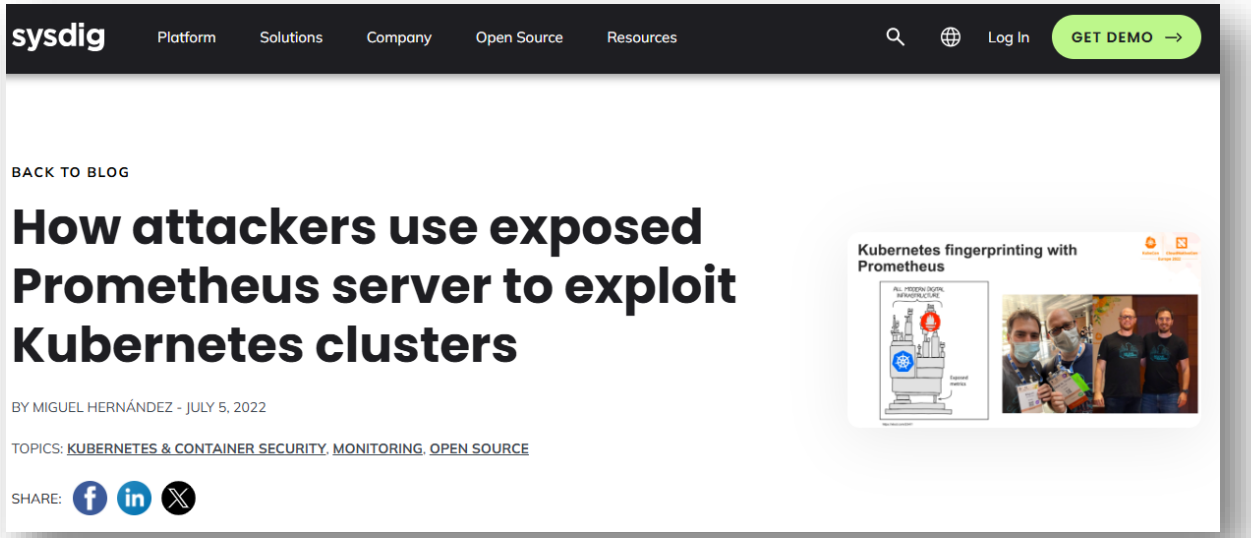
Prometheus: A Case Study



Exposed Application + **Anonymous Access** = **Security Risk**

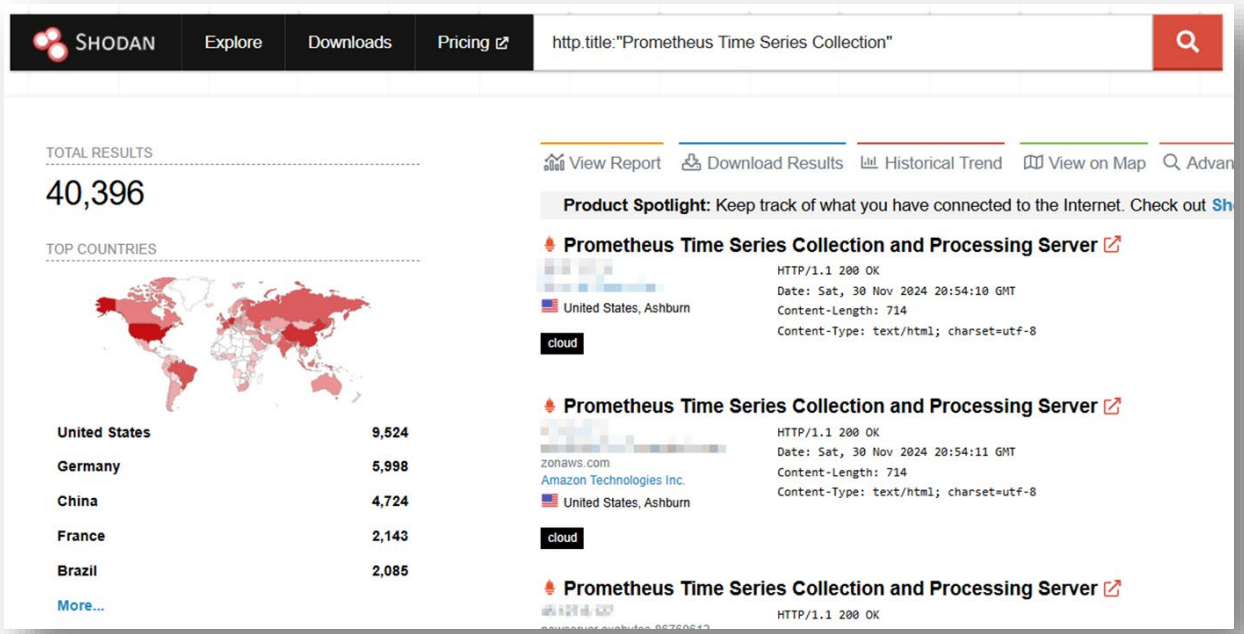
Information Disclosure Through Prometheus

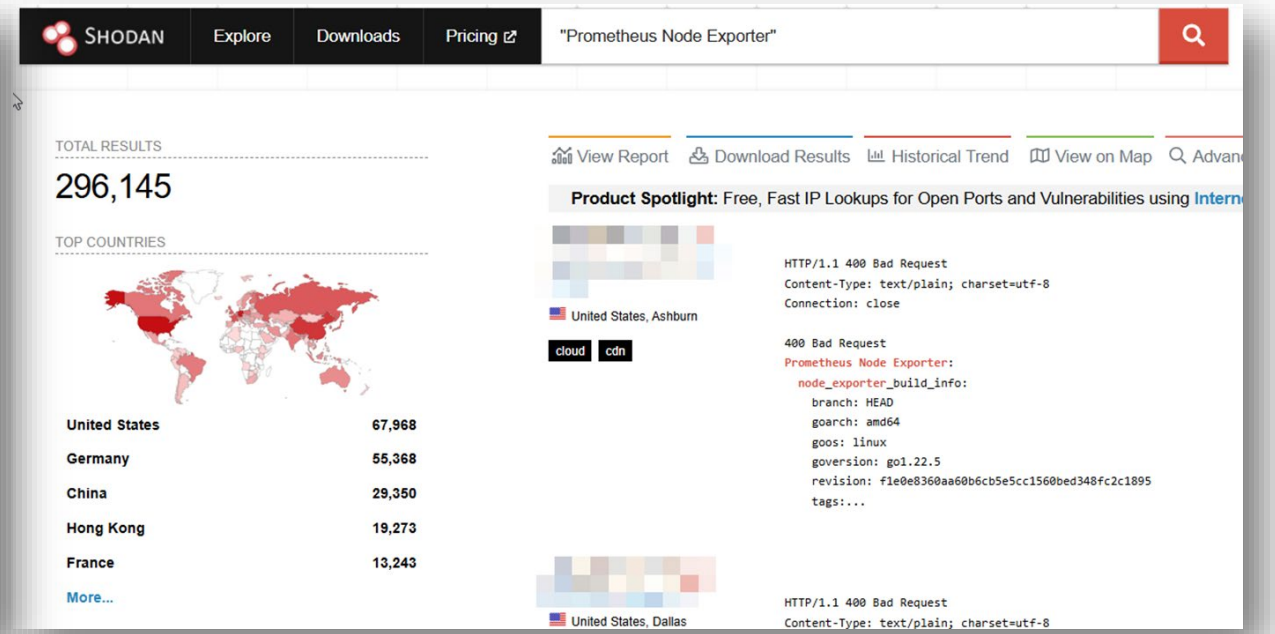




https://jfrog.com/blog/dont-let-prometheus-steal-your-fire/
https://sysdig.com/blog/exposed-prometheus-exploit-kubernetes-kubeconeu/

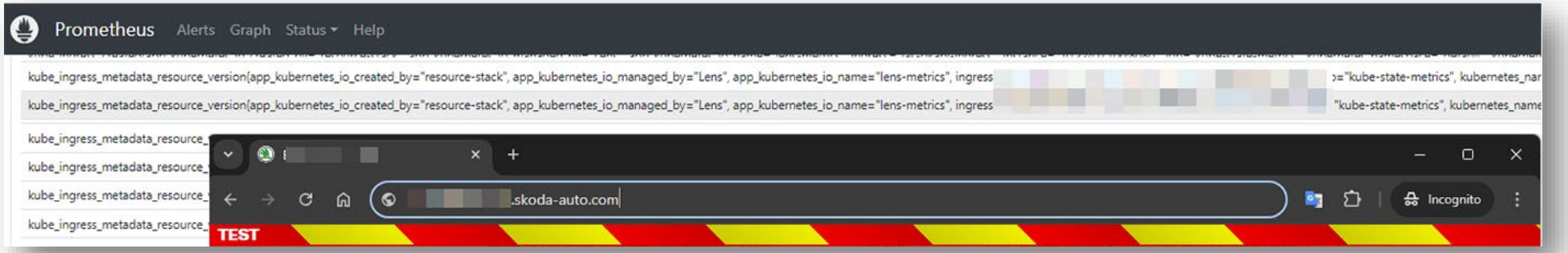
Shodan & Prometheus





300,000+

Our Findings

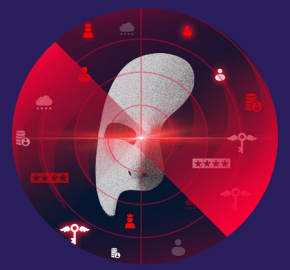


Mitigations for Secrets in Applications

Enable Authentication & Authorization: Restrict access to sensitive data and system internals with proper authentication

Limit External Exposure: Minimize public access - use VPNs or private networks when necessary

RSAC™ 2025 Conference

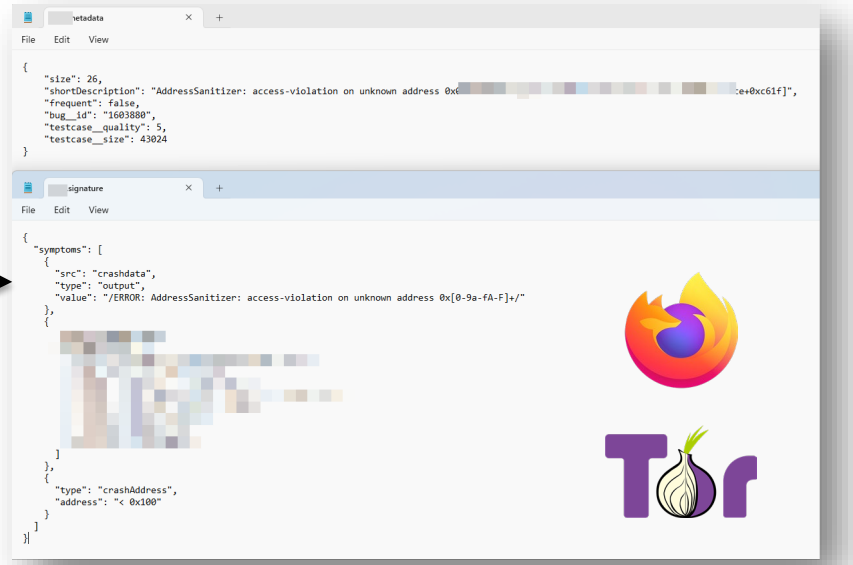


Phantom Secrets

Many Voices.
One Community.

Mozilla FuzzManager API Token

```
+ #!/bin/bash -ex
+
+ cd /home/worker
+
+ git clone -v --depth 1 https://github.com/mozillasecurity/fuzzmanager.git
+ pip install -r fuzzmanager/requirements.txt
+ python fuzzmanager/setup.py install
+
+ cat > $HOME/.fuzzmanagerconf << EOL
+ [Main]
+ serverhost = fuzzmanager.fuzzing.mozilla.org
+ serverport = 443
+ serverproto = https
+ serverauthtoken = [redacted]
+ sigdir = /home/worker/signatures
+ EOL
+ echo "clientid =" `hostname` >> $HOME/.fuzzmanagerconf
```



#2030076

Mozilla FuzzManager API Token Exposed in Git Commit

43 Share:

SUMMARY BY MOZILLA

The researcher has discovered that an API token for the FuzzManager of Mozilla (https://fuzzmanager.fuzzing.mozilla.org) was leaked in one of our GitHub repositories. The API token provides access to our internal fuzzing data and results. The token was accidentally configured with read-write access, we rotated the tokens and made sure to use write-only tokens in our workers

Reported June 17, 2023, 8:25pm UTC

yakirka

Participants

Reported to Mozilla Managed

Report Id #2030076 Resolved

Disclosed November 29, 2023, 10:40am UTC

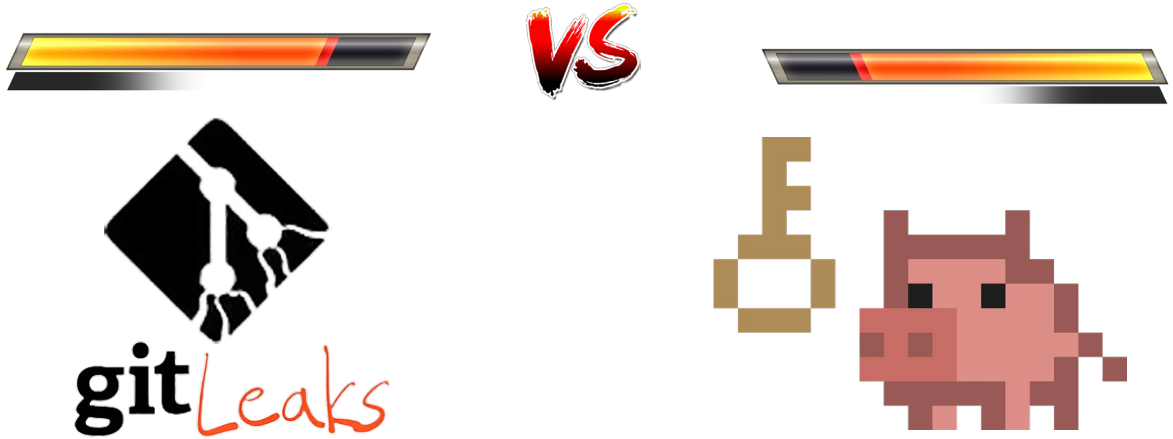
Severity Critical (9 ~ 10)

Popular Strategies for Detecting Secrets

Prevent sensitive data with Git pre-hooks

Scan codebase and history for secrets

Repeat scans regularly



Regex search

Entropy search

Blind Spots of Secrets Scanning Tools

Secrets **don't follow standard regex**

Varying **entropy** levels

Secrets can be in **unexpected places**

We only scan part of our codebase

Our Research

- Secret scanning tools and users often use the `$ git clone`
- Git and SCM edge cases and design choices can cause

`$ git clone` to miss content

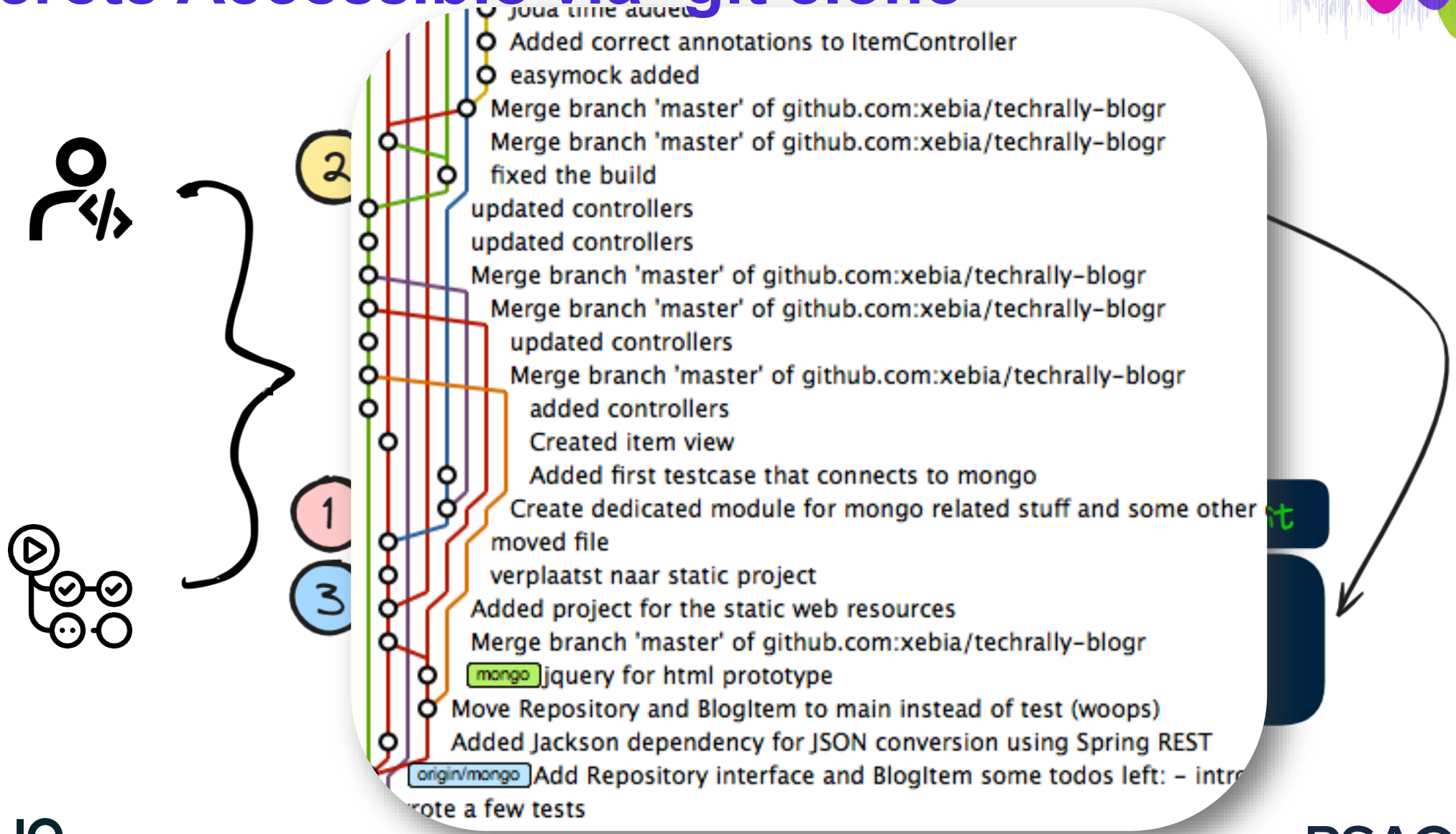
Where Are Secrets Accessible?

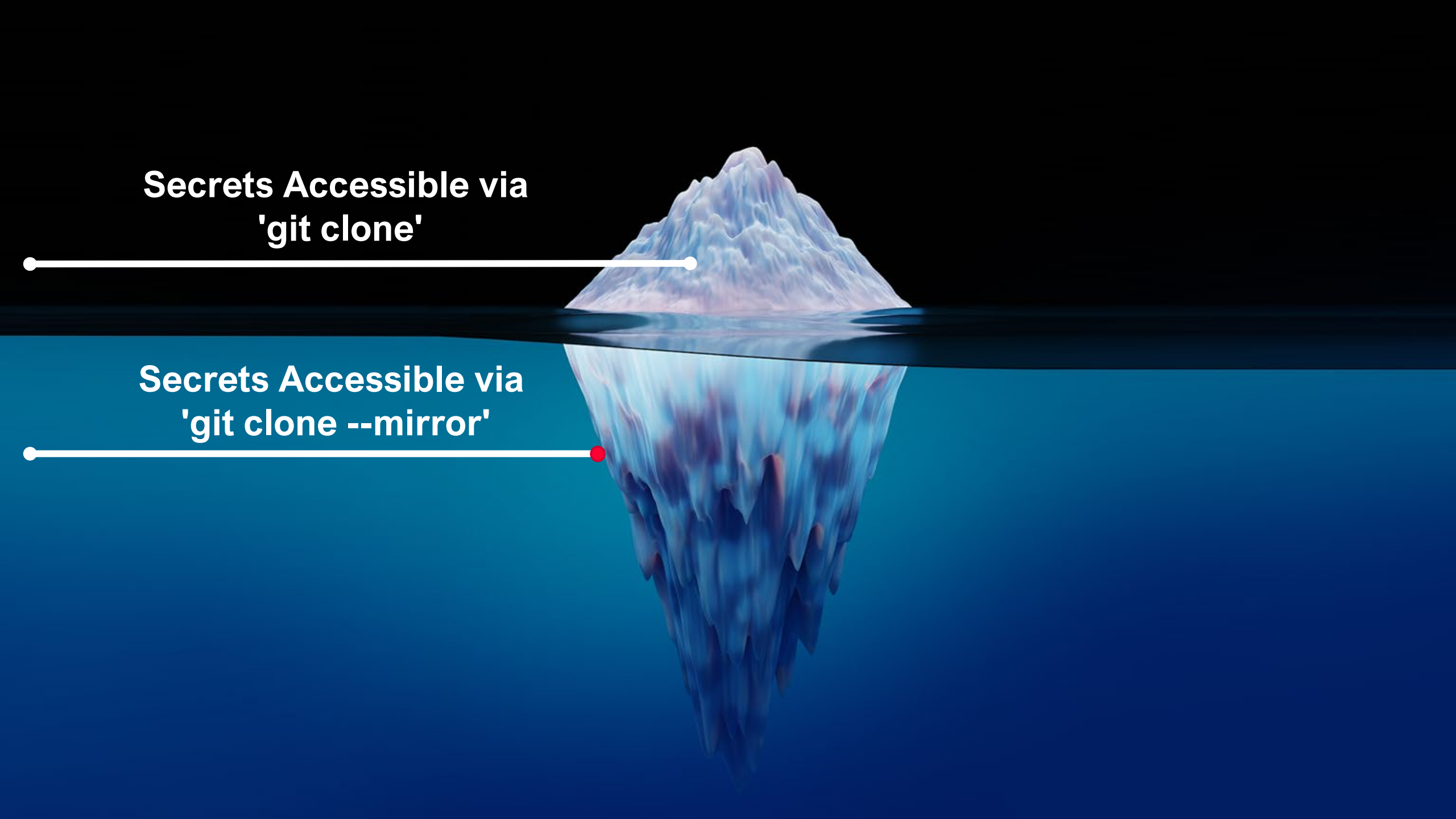
```
$ git clone
```

```
$ git clone --mirror
```

Secrets Accessible via 'git clone'

Secrets Accessible via `git clone`


Added correct annotations to ItemController
easymock added
Merge branch 'master' of github.com:xebia/techrally-blogr
Merge branch 'master' of github.com:xebia/techrally-blogr
fixed the build
updated controllers
updated controllers
Merge branch 'master' of github.com:xebia/techrally-blogr
Merge branch 'master' of github.com:xebia/techrally-blogr
updated controllers
Merge branch 'master' of github.com:xebia/techrally-blogr
added controllers
Created item view
Added first testcase that connects to mongo
Create dedicated module for mongo related stuff and some other
moved file
verplaatst naar static project
Added project for the static web resources
Merge branch 'master' of github.com:xebia/techrally-blogr
mongo jquery for html prototype
Move Repository and BlogItem to main instead of test (woops)
Added Jackson dependency for JSON conversion using Spring REST
origin/mongo Add Repository interface and BlogItem some todos left: - intr
rote a few tests
2
1
3


Secrets Accessible via
'git clone'
Secrets Accessible via
'git clone --mirror'

GitBleed – Finding Secrets in Mirrored Git Repositories – CVE-2022-24975

February 11, 2022 nightwatchcyber Research, Tools git, gitbleed, github, gitlab

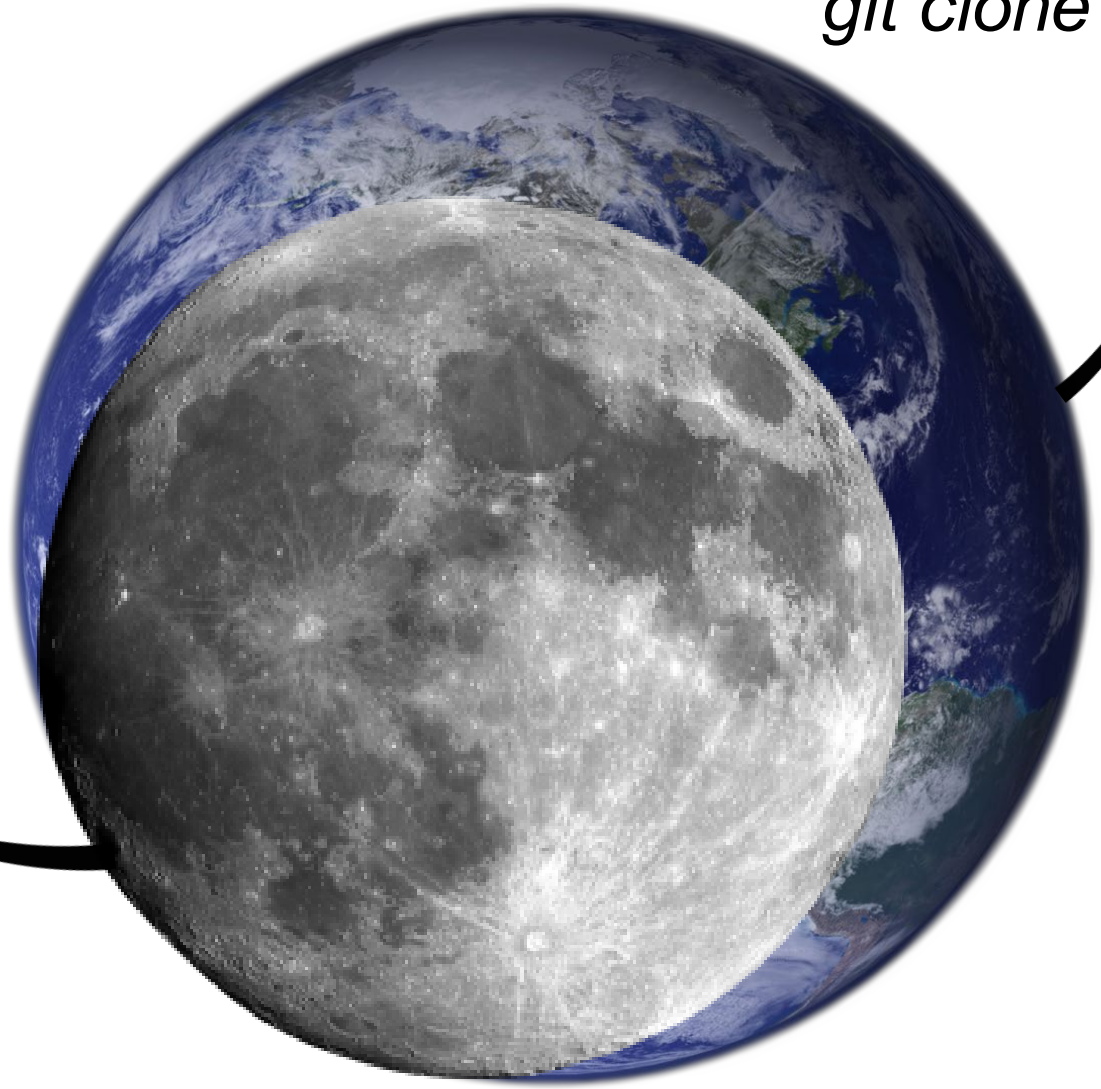
SEARCH

Search ...

CATEGORIES

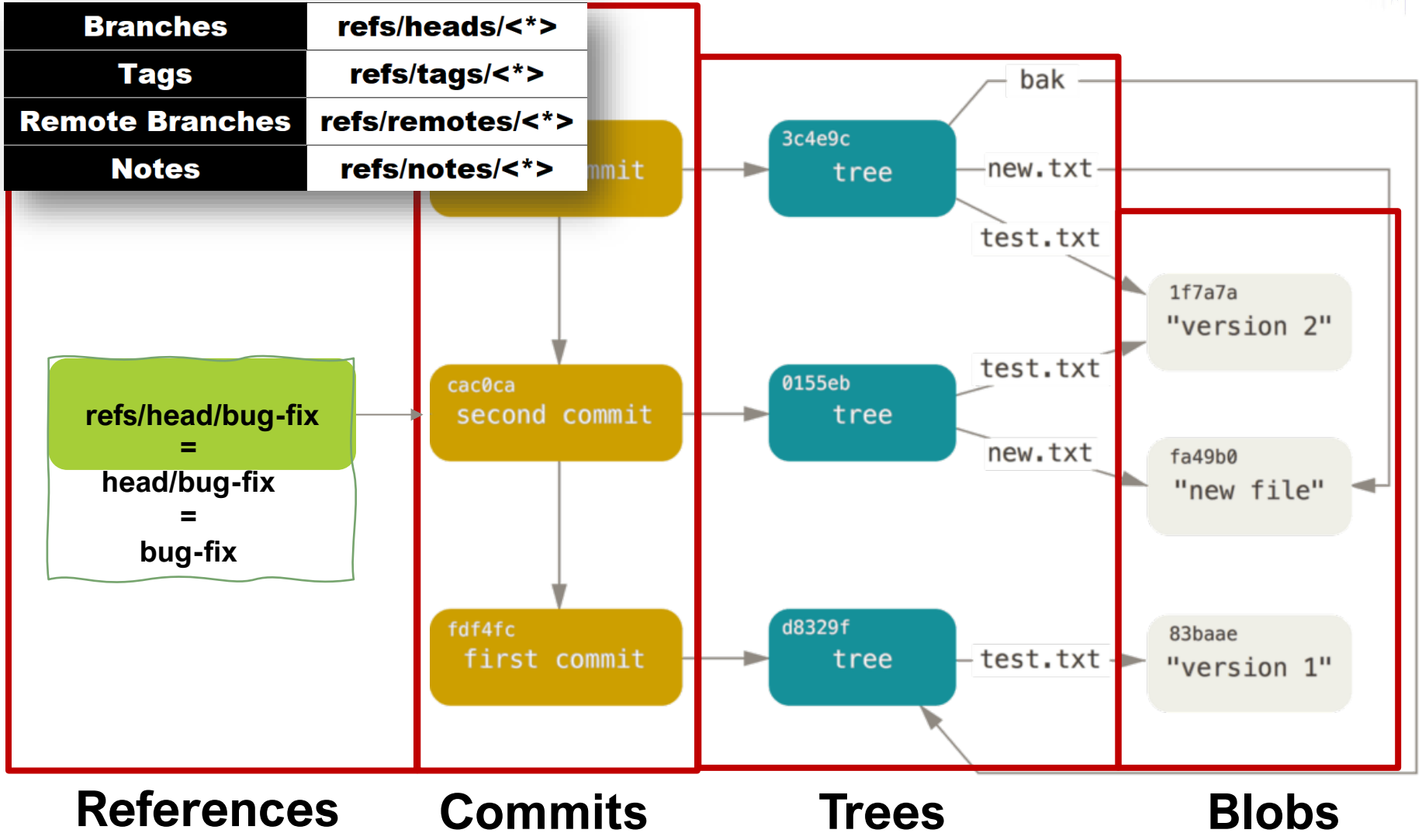
Advisories (66)
Opinion (4)
Research (43)
Tools (4)
Uncategorized (2)

Secrets Within
'git clone --mirror'

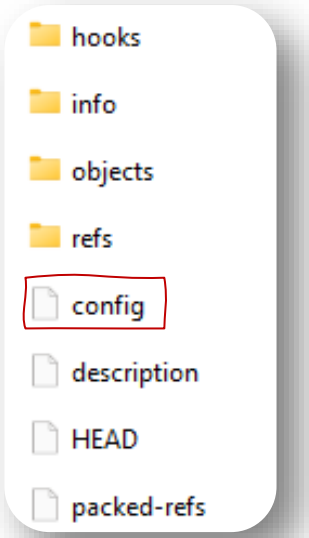


Secrets Within
'git clone'

Git Internals – Commit, Tree, Blob



Git Internals – The `.git/config` File

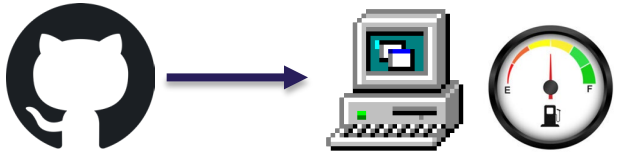


git clone

My computer

```
> git clone https://github.com/yakirk/my-repo.git
> cd my-repo/.git
> cat config
```

```
[remote "origin"]
    url = https://github.com/yakirk/my-repo
    fetch = +refs/heads/* refs/remotes/origin/*
```



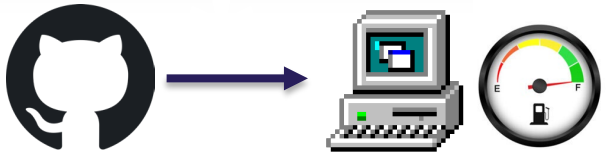
refs/heads/* refs/remotes/origin/*

git clone --mirror

My computer

```
> git clone --mirror https://github.com/yakirk/my-repo.git
> cd my-repo.git
> cat config
```

```
[remote "origin"]
    url = https://github.com/yakirk/my-repo
    fetch = +refs/*:refs/*
```

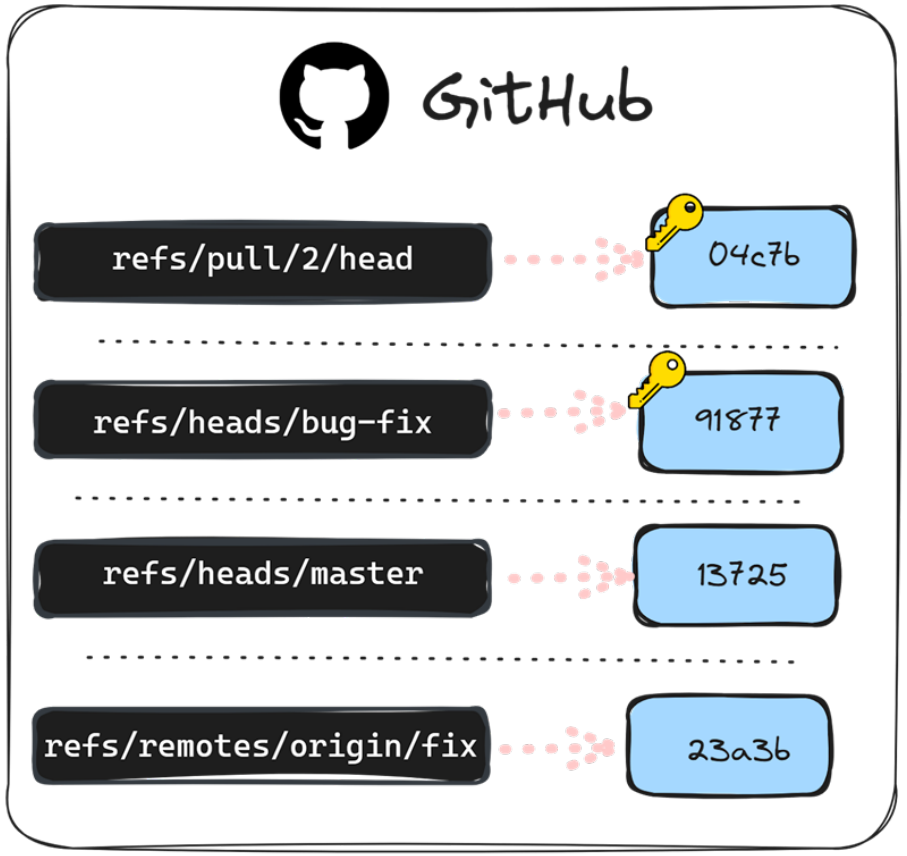


refs/* refs/*

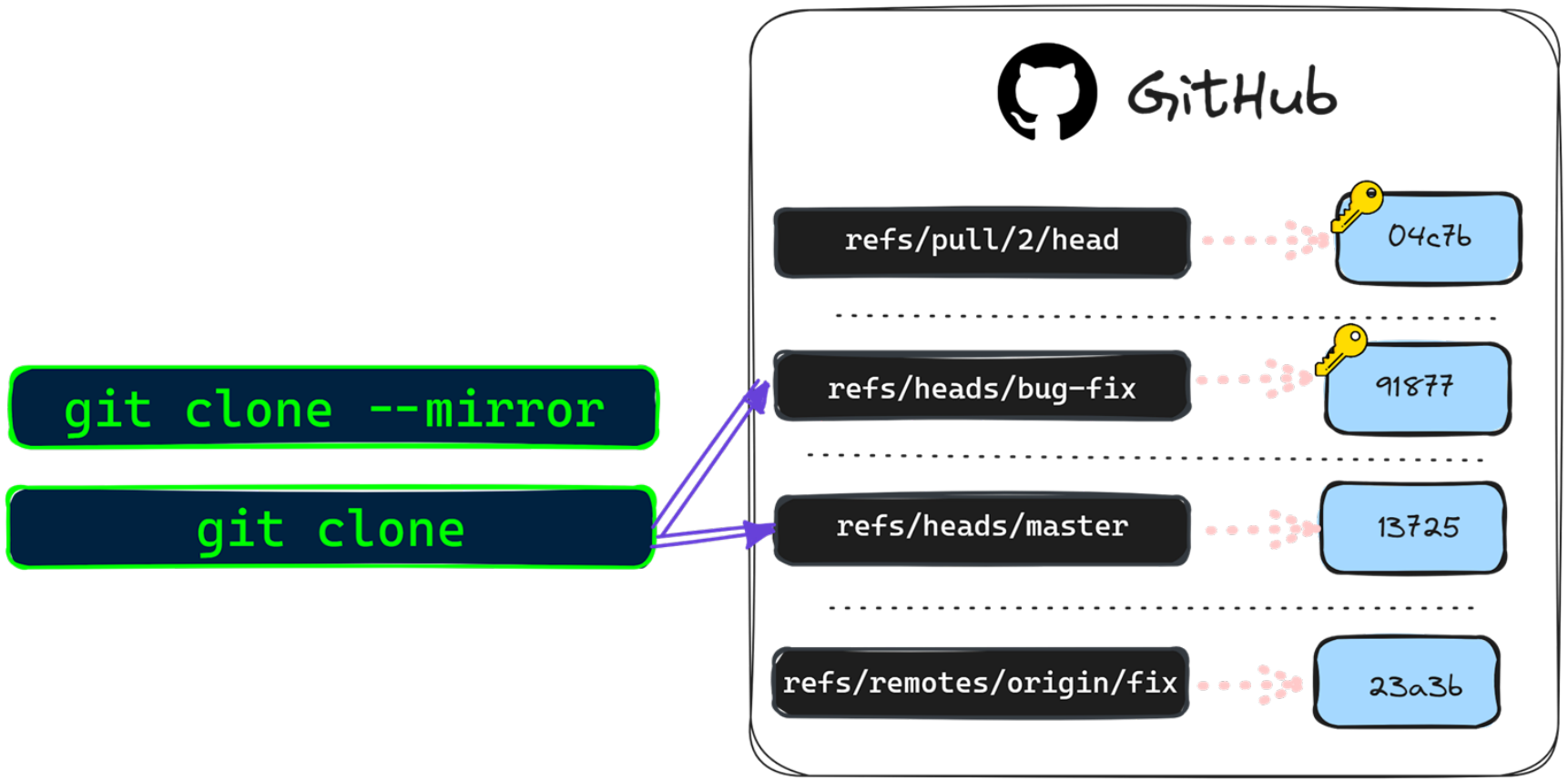
`git clone` vs. `git clone --mirror`

```
git clone --mirror
```

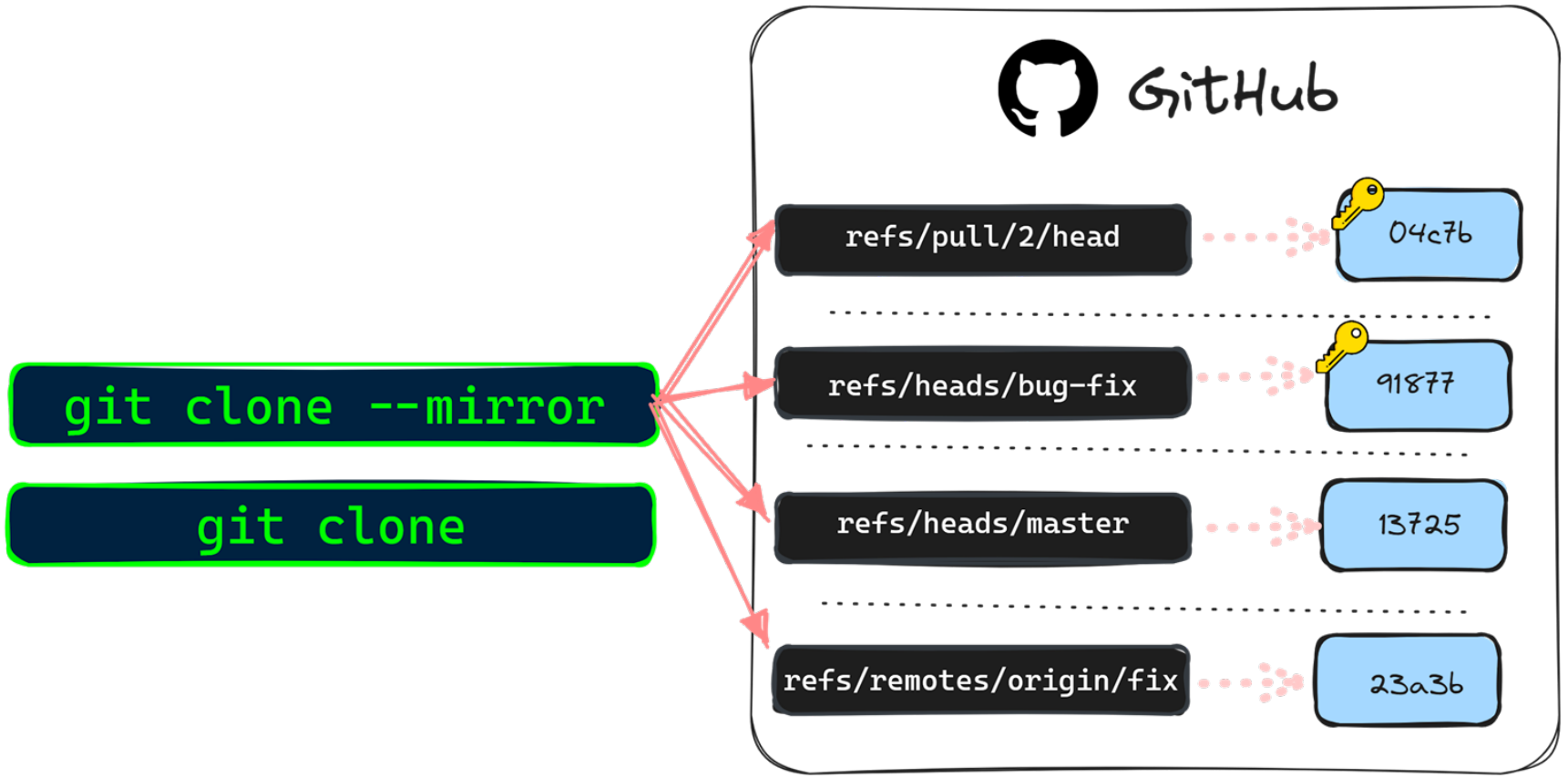
```
git clone
```



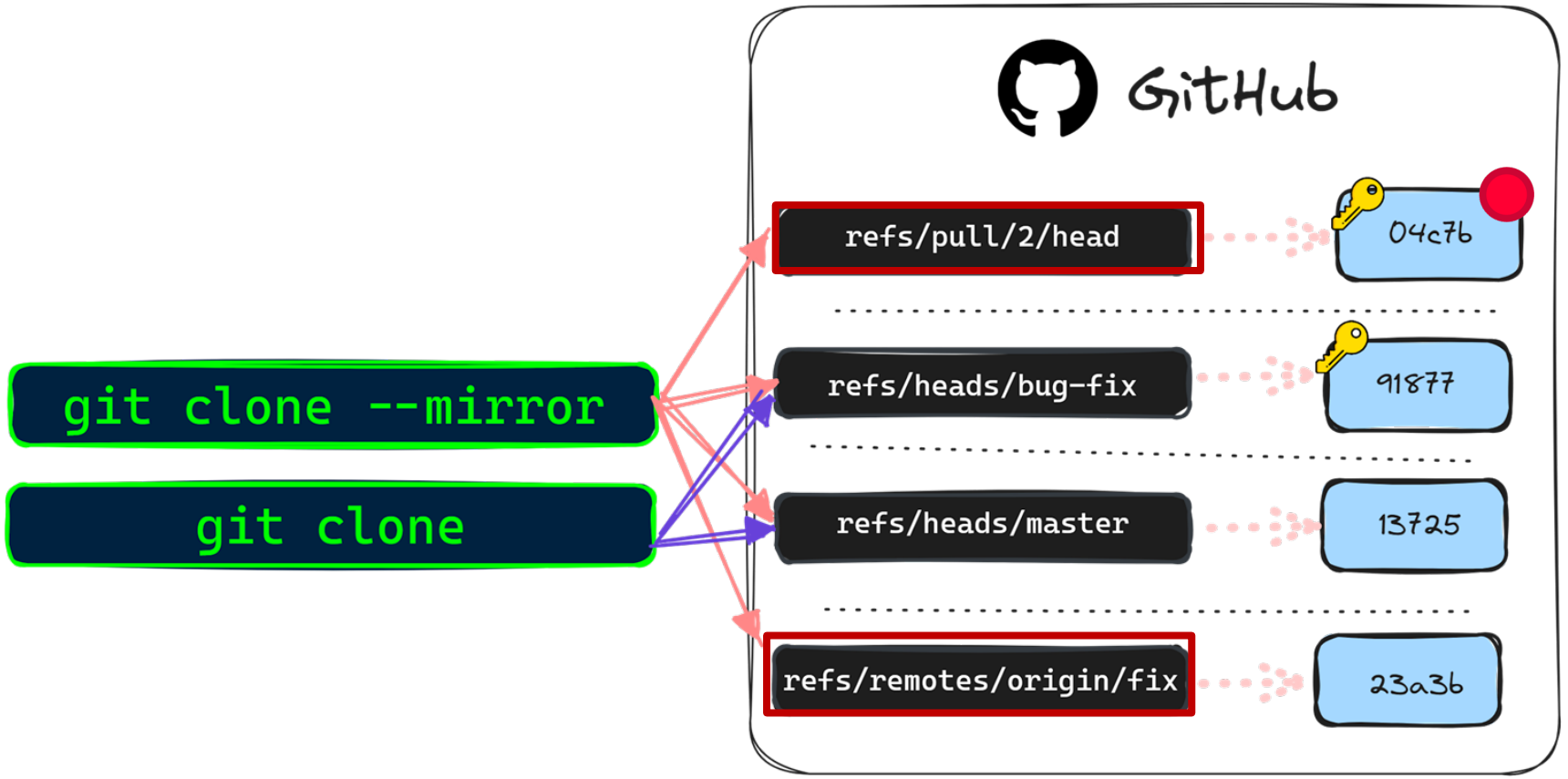
`git clone` vs. `git clone --mirror`



`git clone` vs. `git clone --mirror`



`git clone` vs. `git clone --mirror`



One of Many Examples: PR References

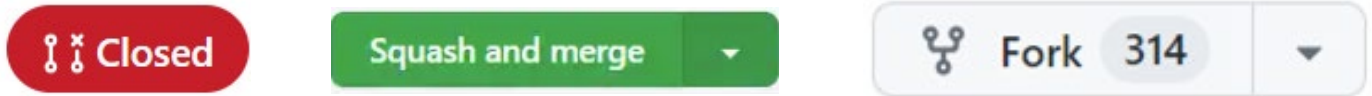
GitHub stores PR info in `*refs/pull/<pull_id>/head*`

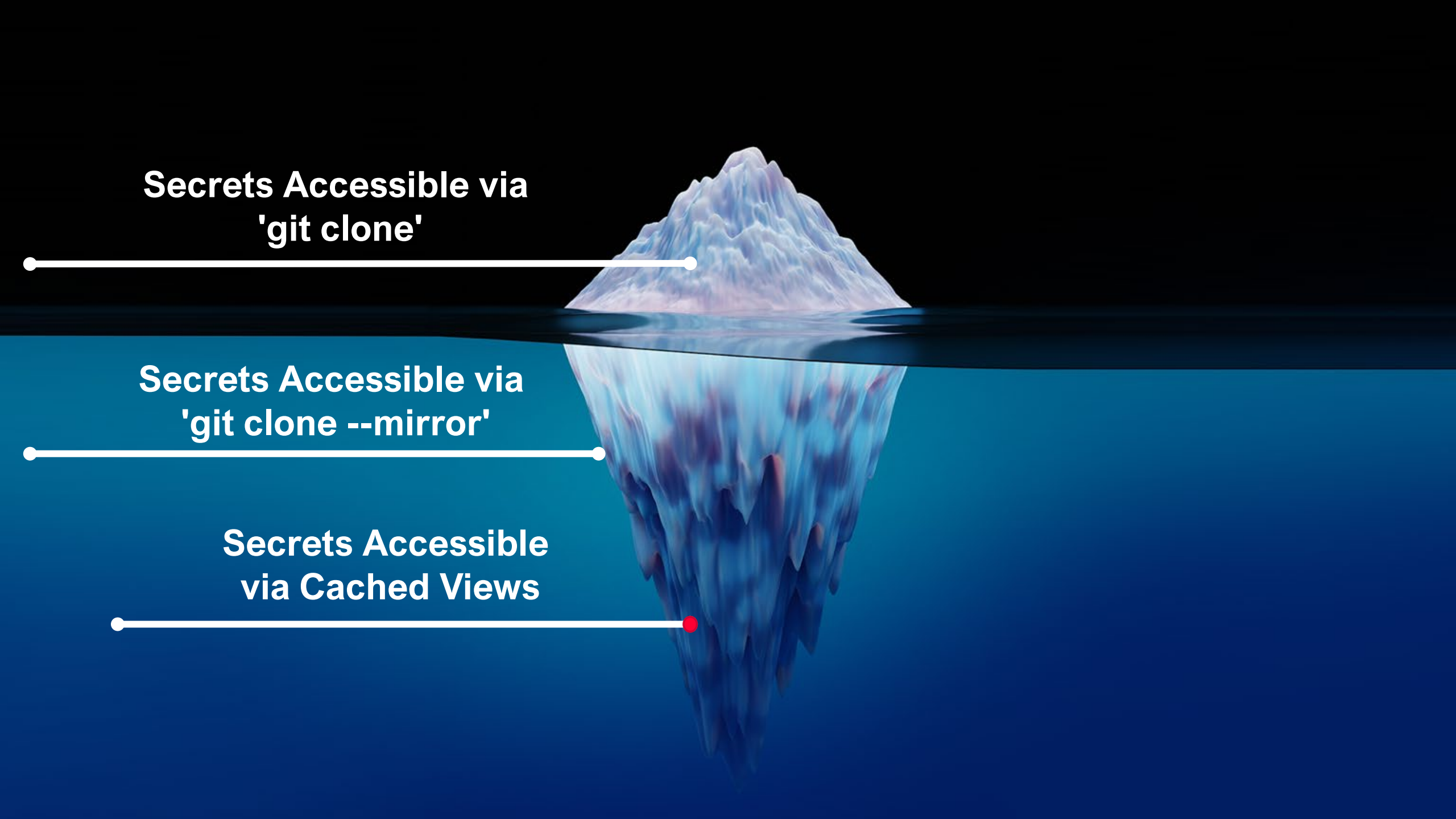
Pull request refs

Branches	refs/heads/<*>
Tags	refs/tags/<*>
Remote Branches	refs/remotes/<*>
Notes	refs/notes/<*>

This reference is fetched **only in a mirrored clone**

```
root@home-virtual-machine:/Random-OSS.git# git ls-remote
From git@github.com:        /Random-OSS.git
f3a34e5d49518419d4f7d5546ed518b55272bb38        HEAD
f3a34e5d49518419d4f7d5546ed518b55272bb38        refs/heads/main
1c329eca9895e1d8bf4fbdc736fb1c6d756715cc        refs/pull/1/head
```



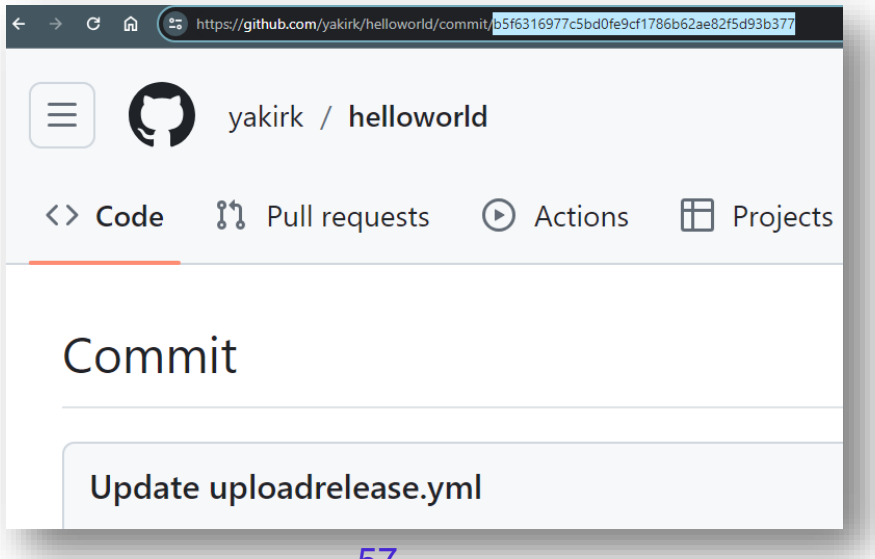

Secrets Accessible via
'git clone'
Secrets Accessible via
'git clone --mirror'
Secrets Accessible
via Cached Views

Secrets Accessible via Cached Views

Everything committed to an SCM platform is **indelible**

Removing a secret-containing commit **doesn't** eliminate access

Knowing the commit hash **allows retrieval** via the SCM API/GUI



Feb 16, 2024 ~9 min read

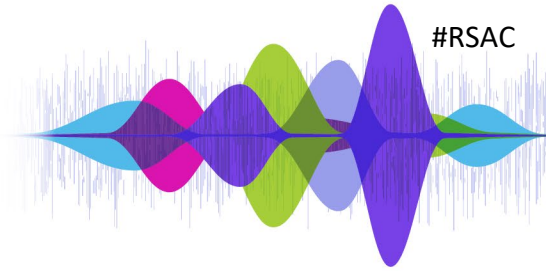
Hidden GitHub Commits and How to Reveal Them

Authored by:
Tobias

TL;DR ¶

We have created a tool for GitHub that can reveal commits that potentially contain sensitive information and are not accessible via the public Git history, but that may be of interest or were intentionally deleted.

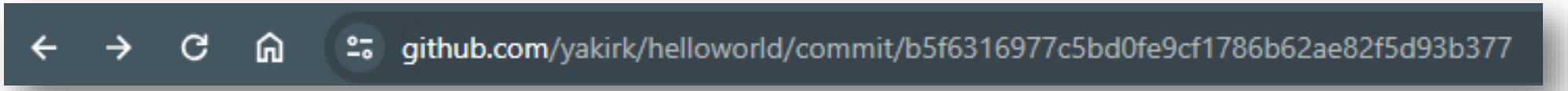
The Challenge With This Approach

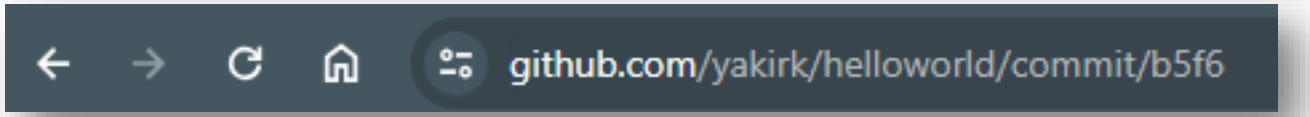


SHA-1

1,461,501,637,330,902,918,203,684,832,716,283,019,655,932,542,976

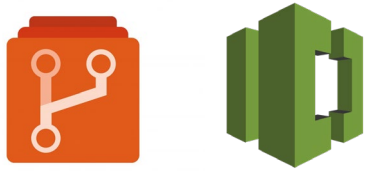
1 - Brute Forcing Cached View Commits





4+

7+



40

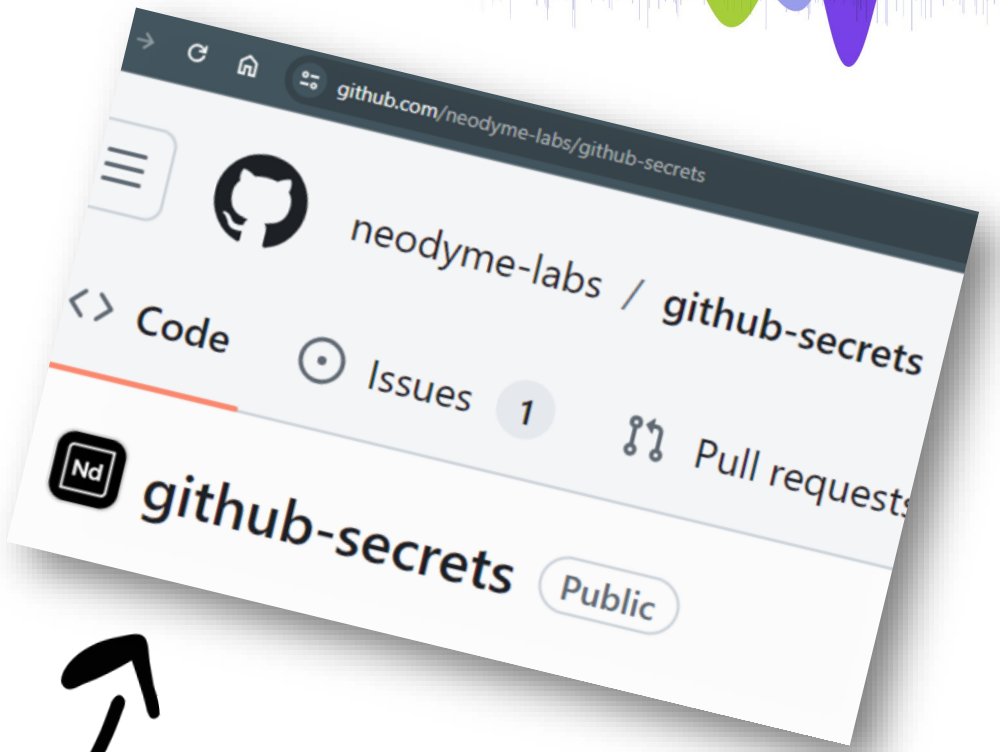
2 - Using REST API Endpoints for Events

REST API / Activity /

REST API endpoints for events

Use the REST API to interact with GitHub events.


```
[
  {
    "id": "22249084964",
    "type": "PushEvent",
    "actor": {
      "id": 583231,
      "login": "octocat",
      "display_login": "octocat",
      "gravatar_id": "",
      "url": "https://HOSTNAME/users/octocat",
      "avatar_url": "https://avatars.githubusercontent.com/u/583231?v=4"
    },
    "repo": {
      "id": 1296269,
      "name": "octocat/Hello-World",
      "url": "https://HOSTNAME/repos/octocat/Hello-World"
    },
    "payload": {
      "push_id": 10115855396,
      "size": 1,
      "distinct_size": 1,
      "ref": "refs/heads/master",
      "head": "7a8f3ac80e2ad2f6842cb86f576d4bfe2c03e300",
      "before": "883efe034920928c47fe18598c01249d1a9fdabd",
      "commits": [
        {
          "sha": "7a8f3ac80e2ad2f6842cb86f576d4bfe2c03e300",
          "author": {
```

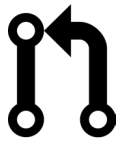



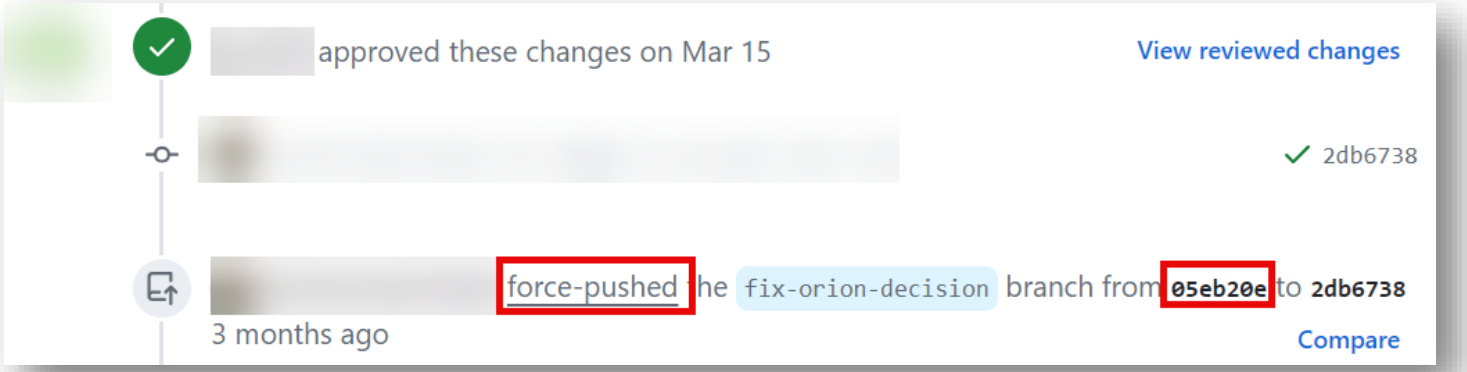
- The last 300 events
- The last 90 days

3 - GUI of Pull Requests

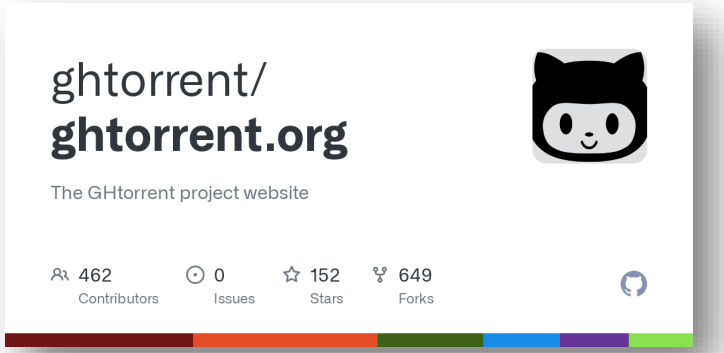
```
$ git checkout –b new-branch
$ git commit –m ‘a mistake/secret’
$ git push origin new-branch
$ git reset --hard
HEAD~{number_of_commits}
$ git push --force
```

```
$ git checkout –b new-branch
$ git commit –m ‘a mistake/secret’
$ git push origin new-branch
$ <edit the file with the secret>
$ git add <new_file> &&  git commit --amend
$ git push --force
```





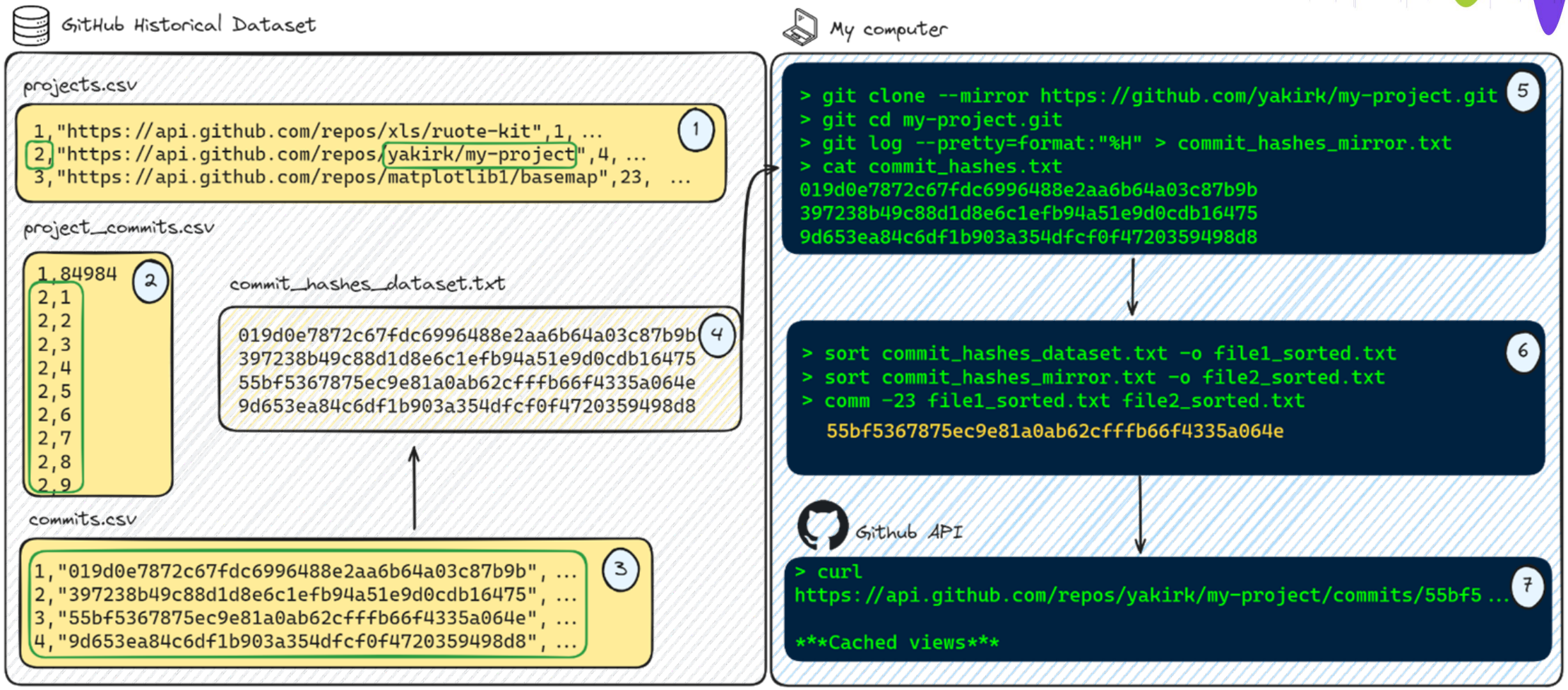
4 - Using GitHub Historical Dataset

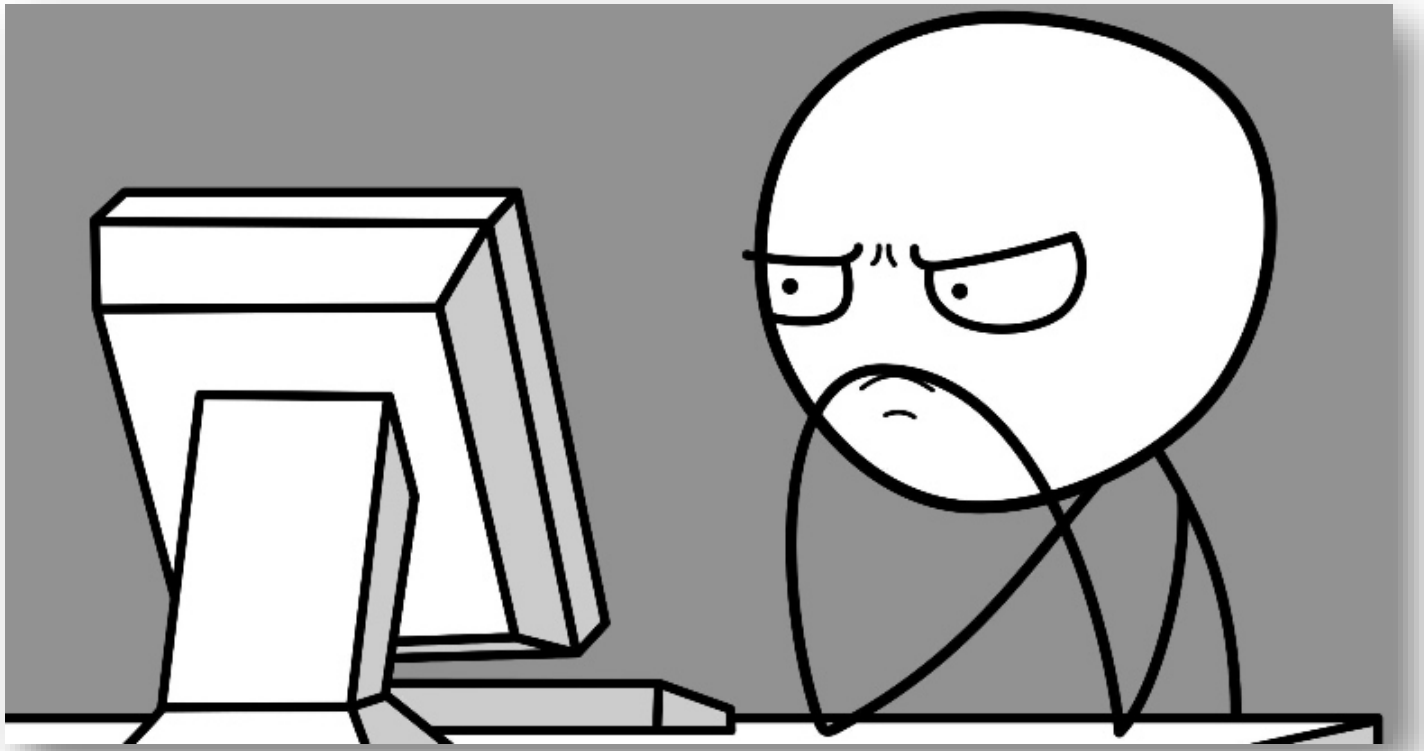


Index of /mysql/

../		
mysql-2013-10-12.sql.gz	10-Dec-2015 20:33	4522065160
mysql-2014-01-02.sql.gz	10-Dec-2015 21:19	5921235276
mysql-2014-04-02.sql.gz	10-Dec-2015 22:13	7354431193
mysql-2014-08-18.sql.gz	10-Dec-2015 23:43	12043734230
mysql-2014-11-10.sql.gz	11-Dec-2015 01:34	15118378692
mysql-2015-01-04.sql.gz	11-Dec-2015 03:42	17389100969
mysql-2015-04-01.sql.gz	11-Dec-2015 06:56	26293878411
mysql-2015-06-18.sql.gz	11-Dec-2015 11:18	35102522985
mysql-2015-08-07.sql.gz	11-Dec-2015 15:17	33069692808
mysql-2015-09-25.tar.gz	11-Dec-2015 20:02	33841191143
mysql-2016-01-08.tar.gz	08-Jan-2016 21:57	35591472888
mysql-2016-01-16.tar.gz	16-Jan-2016 08:17	35838991852
mysql-2016-02-01.tar.gz	01-Feb-2016 11:38	36667951779
mysql-2016-02-16.tar.gz	21-Feb-2016 23:45	37302751172
mysql-2016-03-01.tar.gz	01-Mar-2016 11:57	37988648250
mysql-2016-03-16.tar.gz	16-Mar-2016 10:42	38707567798
mysql-2016-04-19.tar.gz	19-Apr-2016 17:46	40105071925
mysql-2016-05-04.tar.gz	05-May-2016 02:35	40494259095
mysql-2016-06-01.tar.gz	01-Jun-2016 11:50	41787169343
mysql-2016-06-16.tar.gz	16-Jun-2016 11:20	42423227238
mysql-2016-07-19.tar.gz	23-Jul-2016 09:24	43325816626
mysql-2016-09-05.tar.gz	05-Sep-2016 23:18	45284829230
mysql-2017-01-19.tar.gz	20-Jan-2017 04:22	51960147283
mysql-2017-02-01.tar.gz	01-Feb-2017 12:42	52582882424
mysql-2017-03-01.tar.gz	01-Mar-2017 14:38	52916505432
mysql-2017-04-01.tar.gz	01-Apr-2017 14:13	56115975886
mysql-2017-05-01.tar.gz	01-May-2017 14:40	57721654657
mysql-2017-06-01.tar.gz	01-Jun-2017 15:02	59315227769
mysql-2017-07-01.tar.gz	01-Jul-2017 15:05	60948681616
mysql-2017-09-01.tar.gz	01-Sep-2017 15:53	64258782505
mysql-2017-10-01.tar.gz	01-Oct-2017 15:57	65448079781
mysql-2017-12-01.tar.gz	01-Dec-2017 16:49	69797297007
mysql-2018-01-01.tar.gz	01-Jan-2018 16:52	71446490168
mysql-2018-02-01.tar.gz	01-Feb-2018 20:09	73273914729
mysql-2018-03-01.tar.gz	01-Mar-2018 19:13	74476124928
mysql-2018-04-01.tar.gz	01-Apr-2018 18:42	77229267705
mysql-2018-05-01.tar.gz	01-May-2018 07:55	43595377122
mysql-2018-06-01.tar.gz	01-Jun-2018 21:44	81498553814
mysql-2018-07-01.tar.gz	01-Jul-2018 22:13	83433202239
mysql-2018-08-01.tar.gz	02-Aug-2018 03:30	85393483870
mysql-2018-09-01.tar.gz	02-Sep-2018 18:26	87566643663
mysql-2018-10-01.tar.gz	02-Oct-2018 00:59	89297751747
mysql-2018-11-01.tar.gz	02-Nov-2018 00:24	91816085112
mysql-2019-01-01.tar.gz	02-Jan-2019 13:49	96839747312
mysql-2019-02-01.tar.gz	02-Feb-2019 02:13	99195306379
mysql-2019-03-01.tar.gz	02-Mar-2019 02:27	101247922664
mysql-2019-04-01.tar.gz	02-Apr-2019 04:22	103242267957
mysql-2019-05-01.tar.gz	02-May-2019 05:06	105559226039
mysql-2019-06-01.tar.gz	02-Jun-2019 03:38	107975898009
mysql-2020-07-17.tar.gz	18-Jul-2020 08:13	137338062980
mysql-2021-03-06.tar.gz	07-Mar-2021 13:31	156656597501

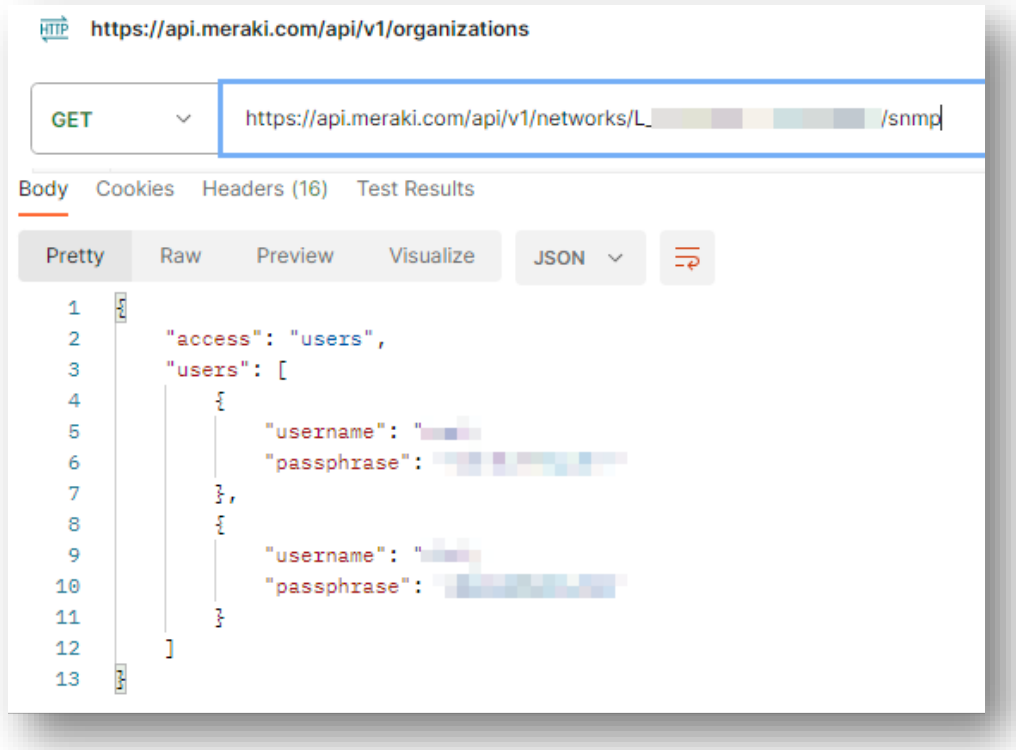
4 - Using GitHub Historical Dataset





Top 100 Organizations
~50,000
17.78%
NVIDIA
aws
airbnb
Uber
Alibaba

Meraki API Tokens of Fortune 500 Companies



cisco Meraki

*The researcher have discovered privileged Meraki API tokens used by **some Fortune 500 companies**. These tokens could allow attackers to **access network devices, Simple Network Management Protocol secrets, camera footage, and more**, serving as an initial foothold for the exposed parties*

Mitigations for Phantom Secrets

Rotate any secret accidentally pushed to SCM

Cached views on SCM platforms **can still access old commits**

Contact GitHub support to delete commits

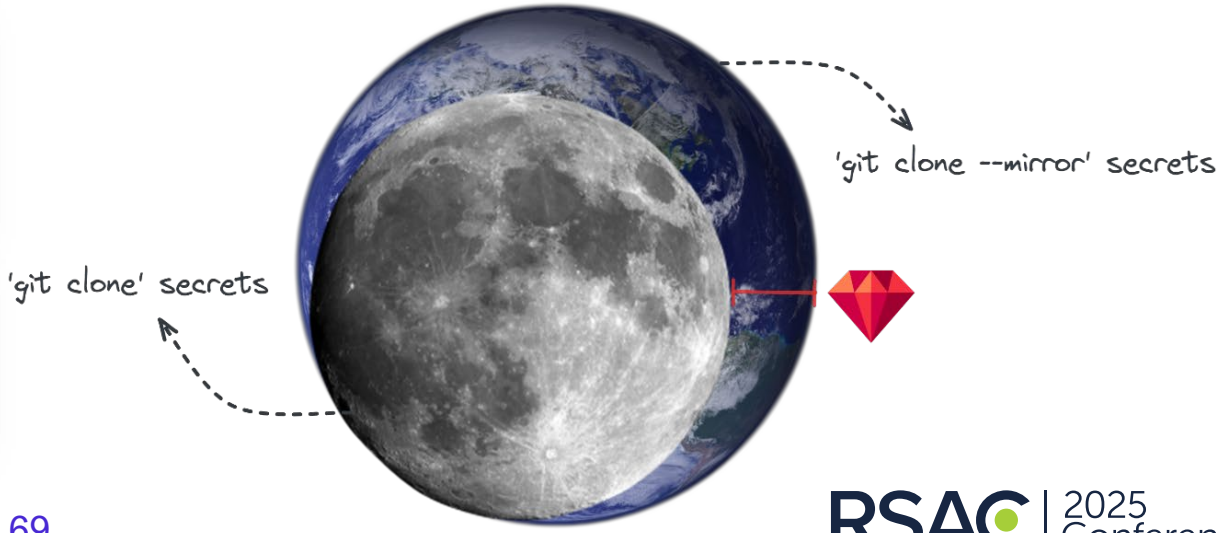
Mitigations for Phantom Secrets

Use `**git clone --mirror**` to scan your repository for secrets	Few secret scanning tools support **bare repositories**	Scan the **delta** between the regular and mirrored clone	Use **GitHub's dataset** or **scan pull requests** for force-pushed commits

gitLeaks

```
git clone --mirror <repository_name>
cd repository_name.git
gitleaks detect .
```

```
git clone --mirror <repository_name>
trufflehog git --bare file://<repository_name>.git/
```



RSAC 2025 Conference

Summary

Mitigations for K8s Secrets

- Most secret scanning tools do not alert or check for encoded secrets
- Scan your codebase for Kubernetes Secrets
- Ensure you encrypt data uploaded to SCM with tools like **Sealed Secrets** or **Mozilla SOPS**

docker-config-secret.yaml

```
rules:
  - id: rule_dockerconfig
    category: general
    title: Dockerconfig secret exposed
    severity: HIGH
    regex: (?i)(\.(dockerconfigjson|dockercfg):\s*\|*\s*(ey|ew)+)
```

```
apiVersion: bitnami.com/v1alpha1
kind: SealedSecret
metadata:
  name: mysecret
  namespace: mynamespace
spec:
  encryptedData:
    foo: AgBy3i4OJSWK+PiTySYZZA9rO43cGDEq.....
```

https://github.com/bitnami-labs/sealed-secrets
https://github.com/getsops/sops

Mitigations for Shadow-IT

- Encourage employees to **scan their personal accounts** for secrets
- **Scan GitHub/Internet** for unique keys and exposed URLs like 'myorg.com/artifacts'
- **Limit keys** to only necessary privileges and scope

Mitigations for Phantom Secrets

- Use `**git clone --mirror**` to scan your repository for secrets
- Few secret scanning tools support **bare repositories**
- Scan the **delta** between the regular and mirrored clone
- Use **GitHub's dataset** or **scan pull requests** for force-pushed commits

```
git clone --mirror <repository_name>
cd repository_name.git
gitleaks detect .
```

```
git clone --mirror <repository_name>
trufflehog git --bare file://<repository_name>.git/
```

'git clone' secrets

'git clone --mirror' secrets

74

Kubernetes Secrets

Secrets in Registries

Secrets in Applications

Phantom Secrets

Mitigations for Secrets in Registries

- Understand that registries play critical role within the SDLC
- Separate your external and internal registries
- Insert registries to your threat modeling practices
- Apply security best practice on all registries

24

Mitigations for Secrets in Applications

- **Enable Authentication & Authorization:** Restrict access to sensitive data and system internals with proper authentication
- **Limit External Exposure:** Minimize public access; use VPNs or private networks when necessary

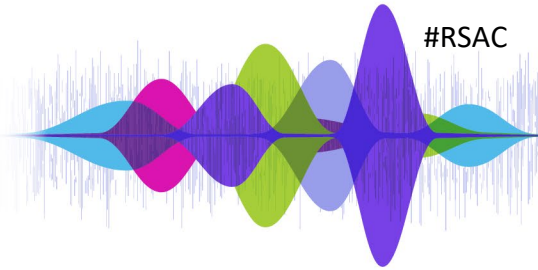
38

RSAC™ 2025 Conference

Apply

Many Voices.
One Community.

Apply What You Have Learned Today



- **Short-Term Actions:**
  - Analyze Codebase Mirrors
  - Monitor Runtime Applications: Detect secret leaks in production.
- **Long-Term Actions:**
  - Track Unique Identifiers: Use techniques like GitHub/Google Dorking to find exposed company assets
  - Encourage employees to scan their personal accounts for secrets

Links

https://www.aquasec.com/blog/the-ticking-supply-chain-attack-bomb-of-exposed-kubernetes-secrets/

https://www.aquasec.com/blog/github-repos-expose-azure-and-red-hat-secrets/

https://www.aquasec.com/blog/250m-artifacts-exposed-via-misconfigured-registries/

https://www.aquasec.com/blog/undetected-hard-code-secrets-expose-corporations/

https://www.aquasec.com/blog/300000-prometheus-servers-and-exporters-exposed-to-dos-attacks/

Shadows in the Cloud: The Invisible Pathways to Breaching AWS Accounts - [SBV-T09]

Tuesday, Apr 29 | 1:15 PM - 2:05 PM PDT

Imagine taking over an entire AWS account with just an account ID. This session will reveal attack vectors that led to the discovery of multiple critical vulnerabilities in AWS, exposing paths to remote code execution, account takeovers, and data leaks. Attendees will learn how issues were uncovered, how to detect them in their own environments, and actionable steps to enhance cloud security.

Session Participant(s)

Yakir Kadkoda
Director of Security Research, Aqua Security

Assaf Morag
Director of Threat Intelligence, Aqua Security

Topic/Track: Sandbox Villages

Type/Format: Track Session

Pass Requirement: On Demand, Speaker, ESAF, Media: Press/Analyst, Media: Content Creator , Cyber Leaders Forum, CISO Boot Camp, Expo Plus, Full Conference

Session Classification: Intermediate —Technical

RSAC™ 2025 Conference

Many Voices.
One Community.

Thank you

@YakirKad
@MoragAssaf